Temporal Information Extraction
Extracting Events and Temporal Expressions
A Literature Survey

Naman Gupta
133050012

June 18, 2015

Chapter 1

Introduction

Information on web is increasing at infinitum. There exists plethora of data on World Wide Web (WWW) in various electronic and digital form. Thus, web has become an unstructured global area where information even if available, cannot be directly used for desired applications. One is often faced with an *information overload* and demands for some automated help. Information extraction (IE) is the task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents by means of Text Mining and Natural Language Processing (NLP) techniques. Extracted structured information can be used for variety of enterprise or personal level task of varying complexity. In this survey report we will discuss literature related to temporal expressions and event extraction.

1.1 Temporal Expression Recognition

Temporal Expression Recognition (TER) is the process of locating phrases that denote temporal information. Temporal expressions may be an expressed point in time, a duration or a frequency. These expressions can be used in information extraction and question-answering to (a) answer time-specific queries, (b) arrange information in a chronological manner, *etc.* Research in TER mostly exists in news domain text, arguably because of availability of large corpora and presence of temporal expressions in news documents. In recent times, TER has also been applied to other domains like medical (Sohn et al., 2013), (Jindal and Roth, 2013), (Xu et al., 2013), (Roberts et al., 2013). Initially, temporal expressions were considered a type of named entities and their identification was part of the named entity recognition task. Since the Automatic Content Extraction program[1] in 2004 there has been a separate task identified and called Temporal Expression Recognition and Normalisation (TERN).

Timex evaluation is now evaluated in two major temporal annotation challenges: TempEval[2] and i2b2[3], both of which prefer the TimeML-level TIMEX3 standard. Recently, Temporal Expression Recog-

[1] http://www.nist.gov/speech/tests/ace

[2] https://www.cs.york.ac.uk/semeval-2013/task1/

[3] https://www.i2b2.org/

nition was a track in Clinical TempEval of SemEval 2015[4] challenge. It aims at extracting temporal information from clinical notes and pathology reports for cancer patients from the Mayo Clinic.

1.1.1 Temporal Expressions in Documents

Based on what temporal information expression refers to, there are different temporal expressions. *e.g.*, a point in time or a duration. In addition, there are different realizations of temporal expressions in natural language. Depending on the realization, different types of information are required to determine the normalized meaning of an expression.

Types of Temporal Expressions

- **Date Expressions**: A date expression refers to a point in time of the granularity "day"(*e.g. "June 18, 2015"* or any other coarser granularity, like "month"(*e.g., "June 2015"*) or "year"(*e.g., "2015"*). In other words these can be calendar dates (*e.g. "January 4"*) and other verbal expressions which can be mapped to calendar dates (*e.g. "Last week"*, *"This month"*, *"next Friday"*, or *"this time"*).

- **Time Expression**: A time expression refers to a point in time of any granularity smaller than "day"such as a part of a day (*e.g., "Friday morning"*) or time of a day (*e.g., "3:30 pm"*). In another words TIME is used for specific time points within a day, for instance, *"4.05 AM"*, or can be relative *"20 minutes ago"*.

- **Duration Expression**: A duration expression provides information about the length of an interval *i.e.* The amount of intervening time between the two end-points of a time interval. Example of Duration expressions are *"ten hours"*, *"last 5 months"*.

- **Set/Frequency Expression**: : A set expression refers to the periodical aspect of an event, For *e.g. "every Friday "* , *"thrice a day"*). Medical Documents like discharge summaries have various frequency terms denoted by Latin abbreviations such as, *"tid (thrice a day)"* , *"q4h (every four hours)"*.

1.2 Event Extraction

Event extraction is a sub-problem of Knowledge Extraction which aims to extracts meaningful information called events in the form of situations, occurrences, action, circumstances *etc* from raw text. Events are also categorized into semantic classes based on their temporal behavior. Extraction of events and its semantic classes requires deeper understanding of syntactic and semantic information of the text.

[4] `http://alt.qcri.org/semeval2015/task6/`

1.2.1 Events and its Properties

Events

Literature presents an abstract view of what Events are and there does not exists a proper definition to Events. Ahn (2006) describes events as *Events are undeniably temporal entities.* Piskorski et al. (2007) describes natural disaster or facts as events. Nishihara et al. (2009) describes a triplet of Place, Object and Action as an Event. Pustejovsky et al. (2005) describes events as *Events are a cover term for situations that happen or occur.* Authors have modified definition of event as per their work and domain.

For our work, we define events as:

- **Situations**: Real Word occurrences that happen or occur. For example terms like *war, floods, launch* are events that describe situations
- **Actions**: Process of doing Something. For example terms like *operation, attempt, mission, assistance,offer* denotes action events.
- **States**: Condition or circumstances that someone or something is in at particular time like *believe, kidnapped, sick.*

Moreover,events may be expressed by means of, verbs as in *Army troops* ***positioned*** *in Ladakh region.* Here word *positioned* describes an Action. Noun or noun phrases, for example in the sentence *Apple decided to* ***launch*** *new series of I-phone models*, here word *launch* describes an Occurrence. Adjectives as in *Fibers remained apparently* ***unaffected*** *by the treatment.* Word *unaffected* describes a State.

Characteristics of an Event

The basic characteristics of an event that distinguishes it from non-event word is temporality, which means that events are linked to time in some way or the other. Based on the temporal behavior events are further characterized as being *Static or Dynamic, Durative or Punctual, Telic or Atelic.*

- Based on the duration of events, they can be Durative or Punctual. Durative events like *desired* lasts for a period of time whereas Punctual events like *throw* are instantaneous in nature.
- Based on change an event bring in the duration of its existence, it can be Static or Dynamic. Static events remain unchanged for their duration, and have very little internal structure. For example *love*, if one loves someone for a period of time, then one loves that person at every point within that time whereas Dynamic events include some sort of change in the world and possesses internal structure. For example, *run* is an event that involves change in the body movements.
- Based on whether events involves notion of natural completion, they can be Telic or Atelic. While Telic events like *found,achieved* include in their meaning a natural completion. Atelic events like *walkig, driving* does not

Chapter 2

Approaches to Temporal Expression Recognition

In this chapter, we will look into some of the methods to Temporal Information Recognition.

2.1 Rule-based Methods

Rule based or knowledge-driven methods exploits human knowledge about the contents of the text to be processed in addition to linguistic and lexicographic knowledge . This knowledge is encoded in the form of patterns that express rules which are used to extract desired information from the text. Information is mined from corpora by using predefined or discovered linguistic patterns, which can be either lexico-syntactic patterns or lexico-semantic patterns. While lexico-syntactic exploits lexical representation with syntactical information, lexico-semantic uses semantic or meaning of the information along with lexical representations.

Heideltime

HeidelTime (Strötgen and Gertz, 2010) is a multilingual, cross-domain temporal tagger developed at the Database Systems Research Group at Heidelberg University. It extracts temporal expressions from documents and normalizes them according to the TIMEX3 annotation standard. HeidelTime distinguishes between news-style documents and narrative-style documents (*e.g.*, Wikipedia articles) in all languages. In addition, English colloquial (*e.g.*, Tweets and SMS) and scientific articles (*e.g.*, clinical trails) are supported.

HeidelTime is developed as a UIMA component. UIMA is widely used for processing unstructured content such as audio, images, or text. Different components can be combined to create a pipeline of modular tools, and all components use the same data structure, the Common Analysis Structure (CAS). This allows to combine tools that were not originally built to be used together, an advantage we are using for preprocessing tasks as well.

Every temporal expression te can be viewed as a three-tuple $te_i = \langle e_i, t_i, v_i \rangle$, where e_i is the expression itself as it occurs in the textual document, t_i represents the type of the expression, and v_i is the normalized value. There are four possible types, namely Date, Time, Duration, and Set. The extraction rules mainly consist of regular expression patterns. However, other features can be used as well, *e.g.*, a constraint what part-of-speech the previous or next token has to have.

Heideltime contains 3 types of resources : Rule resources, pattern resources and normalization resources.For example, a rule to detect expression like "November 2001"following rule is written in the rule resources file :

```
RULENAME="date_r7a",EXTRACTION="(%reMonthLong|%reMonthShort)( of | )
%reYear4Digit", NORM_VALUE="group(5)-%normMonth(group(1))"
```

Here, *RULENAME* is an unique name given to a rule, *Extraction* is the regular expression pattern to extract desired expression. It make use of *reMonthLong*, *reMonthShort*, *reYear4Digit* as pattern resources. *NORM_VALUE* is the normalization Expression which make use of *normMonth* is the normalization resource.

Example of Normalization Resource :

```
normMonth("June") = "06"
normSeason("summer") = "SU"
```

Heideltime achieved F-Score of 90.30% in SemEval 2013 sub-task of Temporal Expression Extraction.

SUTime

SUTIME (Chang and Manning, 2012) is a temporal tagger for recognizing and normalizing temporal expressions in English text. SUTIME is available as a Java Library and is a part of the Stanford CoreNLP pipeline. It can be used to annotate documents with temporal information. It is a deterministic rule-based system designed for extensibility.

SUTIME (i) builds up patterns over individual words to find numerical expressions;then (ii) using patterns over words and numerical expressions to find simple temporal expressions; and finally (iii) forming composite patterns over the discovered temporal expressions. It main features are :

- **Extraction** of temporal expressions from text: Given tokenized English text, SUTIME finds temporal expressions and outputs annotations for further manipulation and interpretation. Its output includes annotations in the form of TIMEX3 tags.
- **Representation** of temporal objects as Java classes: SUTIME provides tools to map them to logical representations and data structures that are easier to handle programmatically.

– **Resolution** of temporal expressions with respect to a reference date: When processing natural language text, one often has to work with expressions that refer to a relative time (*e.g.*, last Friday). Determining the actual date to which such expressions refer requires a reference date, on which the statement was made. SUTIME uses document dates as references.

MedTime

MedTime (Lin et al., 2013) is temporal information extraction system for clinical narratives. It was developed as a part of EVENT/TIMEX3 track of the 2012 i2b2 clinical temporal relations challenge. MedTime uses hybrid system which uses cascaded rule-based and Machine Learning Techniques to extract Event and Time Expressions. It also normalizes extracted Time Expressions. Temporal Tagging was done by incorporating Heideltime temporal tagger. Rule specific to clinical discharge summaries were added to existing rule set. MedTime developed its own *Clinical FREQUENCY TE tagging* algorithm for normalizing time expressions denoting frequency terms like *BID* (twice a day) or *q.8.h* (every 8 hours). ATT system achieved F1-score of 88.0% in i2b2 2012 temporal relations challenge task of Time Expression Extraction.

Other Rule-Based Systems

– **TempEx**: TempEx Mani and Wilson (2000) is one of the first temporal taggers for extracting and normalizing temporal expressions. It is a simple, rule-based system with limited normalization functionality.It annotates document using TIMEX2 tags.

– **GUTime**: A Perl temporal tagger provided by Georgetown University which is based on TempEx. It was developed as reference tool for TimeML using TIMEX3 tags. GUTime was one of the most widely used temporal taggers. It is part of the TARSQI toolkit Verhagen and Pustejovsky (2008) consisting of components for the extraction of events, temporal expressions, and temporal relations . GUTime has been evaluated on the ACE TERN 2004 training data and achieves competitive results.

– **Chronus**: Developed by (Negri and Marseglia, 2004) as a part of ACE TERN 2004 competition to extract and normalize time expressions. Detection/Extraction and normalization part were developed as different component. Extraction component is a rule-based system having around 1000 hand crafted rules. They detect all possible temporal expressions, determine their extent, and gather contextual information relevant for the normalization task.

  Normalization component sets the values of all TIMEX2 attributes based on the context information collected during the detection phase. Either the document creation time (*e.g.*, for "today", "December", "next month") or the previously mentioned expression with a compatible granularity (*e.g.*, for "the following month", "two years ago") is selected depending on the information about the expression gathered during the detection phase.

– **MayoTime**: A Temporal tagger developed by (Sohn et al., 2013). It was adapted from publicly available Temporal Tagger, Heideltime. It finds temporal expression through pattern

matching rules based on regular expression. It is designed in a way that is provides an easy interface for users to customize the system. By achieving F1-score of 90.03% it became the best performing system in i2b2, 2012 temporal relations challenge task of Temporal Expression Recognition.

2.2 Statistical methods

Statistical methods are class of algorithm that learn a model by looking at annotated training examples. Among the supervised learning algorithms for NER, considerable work has been done using Hidden Markov Model (HMM), Decision Trees, Maximum Entropy Models (ME), Support Vector Machines (SVM) and Conditional Random Fields(CRF). Typically, supervised methods either learn disambiguation rules based on discriminative features or try to learn the parameter of assumed distribution that maximizes the likelihood of training data. We will study each of these methods in detail in next sections.

2.2.1 Hidden Markov Models

HMM is the earliest model applied for solving problem related to tagging like Named Entity Recognition (NER), Part-of-speech tagging, Temporal Tagging *etc.* BikelBikel et al. (1999) introduced a system, *IdentiFinder*, to detect NER among which Temporal Expression Recognition was sub-task.

According to Bikel's formulation of the problem in the *Identifinder* system, only a single label can be assigned to a word in context. Therefore, the model assigns to every word, either one of the desired classes or the label NOT-A-NAME to represent "none of the desired classes". State diagram for his model is shown in Figure-2.1

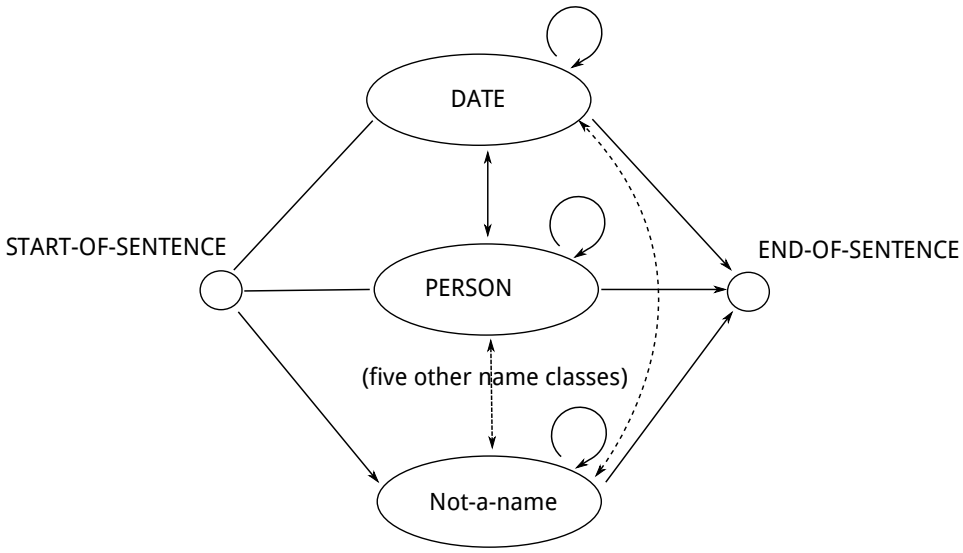

Figure 2.1: State Diagram for Identifier

For tagging a sentence, the task is to find the most likely sequence of name-classes(NC) given a sequence of words(W):

$$\max Pr(NC|W)$$

HMM is a generative model, *i.e.* it tries to generate the data, sequences of words W, and labels NC from distribution parameters.

$$Pr(NC|W) = \frac{Pr(W, NC)}{Pr(W)}$$

The Viterbi algorithm Forney (1973) is used to maximize $Pr(W, NC)$ through the entire space of all possible name-class assignments. Bikel modeled the generation in three steps:

- Select a name-class nc, conditioned on the previous name-class and previous word.
- Generate the first word inside the name-class, conditioning on the current and previous name-classes.

$$Pr(nc|nc_{-1}, w_{-1}).Pr(< w, f >_{first} |nc, nc_{-1})$$

- Generate all subsequent words inside the current name-class, where each subsequent word is conditioned on its immediate predecessor

$$Pr(< w, f > | < w, f >_{-1}, nc)$$

There is also a distinct end marker "+end+", so that the probability may be computed for any current word to be final word of its name-class

$$Pr(< +end+, other > | < w, f >_{final}, nc)$$

Consider an example :

Mr. Jones eats.

Correct annotation for such a sentence is:

Mr. <ENAMEX TYPE="PERSON">Jones</ENAMEX> eats.

Then a max likelihood equation from the search space would be:

```
Pr(NOT-A-NAME | START-OF-SENTENCE, +end+) * Pr("Mr." | NOT-A-NAME, START
-OF-SENTENCE) * Pr(+end+ | "Mr.", NOT-A-NAME) * Pr(PERSON | NOT-A-NAME ,
"Mr.") * Pr("Jones" | PERSON, NOT-A-NAME) * Pr(+end+ | "Jones" PERSON) *
Pr(NOT-A-NAME | PERSON, "Jones") * Pr("eats" | NOT-A-NAME, PERSON) * Pr(
"." | "eats", NOT-A-NAME) * Pr(+end+ | ".", NOT-A-NAME)* Pr(END-OF-SENTE-
NCE | NOT-A-NAME ".")
```

IdentiFinder reported NE accuracy of 94.9% and 90% for a mixed case English (MUC-6 data and a collection of Wall Street Journal documents) and mixed case Spanish (MET-1 data, comprised of articles from news agencies AFP) respectively.

2.2.2 Maximum Entropy based Models

Maximum entropy model, unlike HMM, are discriminative model. Given a set of features and training data, the model directly learns the weight for discriminative features for classification. In Maximum entropy models, objective is to maximize the entropy of the data, so as to generalize as much as possible for the training data. In ME models each feature is associated with parameter λ_i. Conditional probability is thus obtained as follows:

$$P(f|h) = \frac{\prod_i \lambda_i^{g_i(h,f)}}{Z_\lambda(h)}$$
$$Z_\lambda(h) = \sum_f \prod_i \lambda_i^{g_i(h,f)}$$

Maximizing the entropy ensures that for every feature g_i, the expected value of g_i, according to M.E. model will be equal to empirical expectation of g_i in the training corpus.

Finally, Viterbi algorithm is used to find the highest probability path through the trellis of conditional probabilities which produces the required valid tag sequences.

ATT System

Jung and Stent (2013) used big windows and rich syntactic and semantic features for the TempEval time expression and event segmentation and classification tasks. It uses wide variety of features like lexical, Part of speech, dependency and constituency parse and semantic roles.

Problem was modeled as a BIO sequence labeling task. A BIO classifier tags each input token as either Beginning, In, or Out of an time expression. 9 tags namely B-DATE, B-DURATION, B-SET, B-TIME, I-DATE, I-DURATION, I-SET, I-TIME and O were used.

They experimented with context windows of 0, 1, 3, and 7 words preceding and following the token to be labeled (*i.e.* window sizes of (1, 3, 7, and 15).

LLAMA Haffner (2006), machine learning toolkit was used. It encodes multiclass classification problems using binary MaxEnt classifiers to increase the speed of training and to scale the method to large data sets. They also used a front-end to LLAMA that builds unigram, bigram and trigram extended features from basic features; for example, from the basic feature "go there today", it would build the features "go", "there", "today", "go there", "there today", and "go there today".

ATT system achieved F1-score of 85.60% in SemEval 2013 sub-task of Temporal Expression Recognition.

2.2.3 SVM Based Models

Support Vector Machine was first introduced by Cortes and Vapnik (1995) based on the idea of learning a linear hyperplane that separate the positive examples from negative example by large margin. Large margin suggests that the distance between the hyperplane and the point from either instances is maximum. The points closest to hyperplane on either side are known as support vectors.

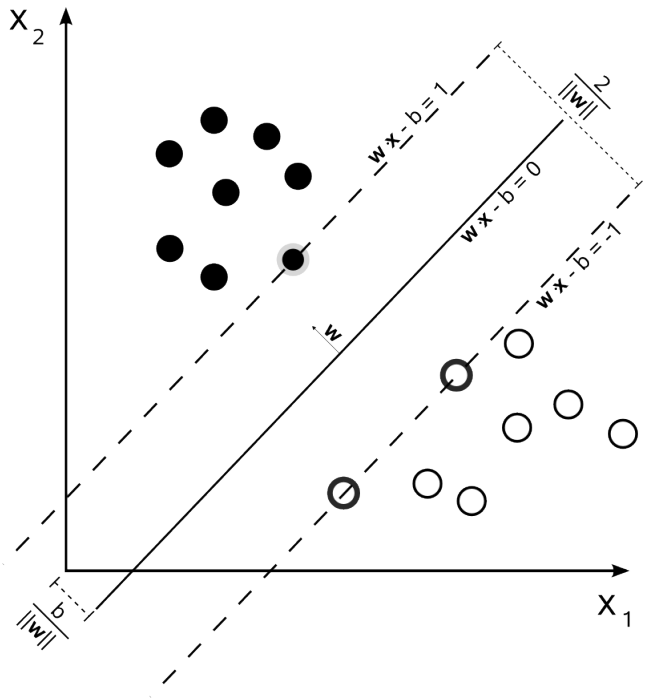

Figure 2.2: Geometric interpretation for SVM

Figure-2.2 shows the geometric interpretation. The linear classifier is based on two parameters, a weight vector W perpendicular to the hyperplane that separates the instances and a bias b which determines the offset of the hyperplane from the origin. A sample x is classified as positive instance if $f(x) = wx + b > 0$ and negative otherwise. If the data points are not linearly separable, then a slack is used to accept some error in classification. This prevents the classifier to overfit the data. When there are more than two classes, a group of classifiers are used to classify the instance.

Bethard (2013) modeled the problem of Temporal Expression Recognition as a BIO token-chunking task, where each token in the text is classified as being at the B(eginning) of, I(nside) of, or O(utside) of a time expression. The Goal of this system was to use a small set of simple features that can be derived from either tokens, part-of-speech tags or syntactic constituency parses. It made use of token's text,stem and POS tag along with the temporal type of each alphanumeric sub-token, derived from a 58-word gazetteer of time words with a window of 3 tokens in both directions.

SVM implementation of LIBLINEAR library was used. Problem was tackled as binary decision problem, *i.e.* if the word belongs to one of the 8 classes, *i.e.* B- Beginning, I- Inside tag for date, time, frequency and duration tags. Thus there are 8 classifiers trained for this purpose.

To produce a single label for each token, the set S of possible tags were identified. If S was empty tag O was assigned else most frequent tag was assigned. If both beginning and inside tags were present then beginning tag was chosen.

2.2.4 CRF Based Models

Conditional random field were introduced by Lafferty et al. (2001) as a statistical modeling tool for pattern recognition and machine learning using structured prediction. Let $o =< o_1, o_2, \cdots, o_T >$ be some observed input data sequence, such as a sequence of words in a text inside the document (the values on n input nodes of the graphical model). Let S be a set of FSM states,each of which is associated with a label, l L, (such as ORG). Let $s =< s_1, s_2, \cdots, s_T >$ be some sequence of states, (the values on T output nodes). By the Hammersley Clifford theorem, CRFs define the conditional probability of a state sequence given an input sequence to be

$$P(s|o) = \frac{1}{Z} \exp\left(\sum_{t=1}^{T} \lambda_k f_k(s_{t-1}, s_t, o, t)\right)$$

where Z is the normalization factor obtained by marginalizing over all state sequences, $f_k(s_{t-1}, s_t, o, t)$ is an arbitrary feature function and λ_k is the learned weight for each feature function. By using dynamic programming, state transition between two CRF states can be efficiently calculated. The modified forward values, $\alpha_T(s_i)$, to be the "unnormalized probability" of arriving state s_i given the observations $< o_1, o_2, \cdots, o_T >$. $\alpha_0(s)$ is set to probability of starting in each state s, and recursively calculated as :

$$\alpha_{t+1}(s) = \sum_{s'} \alpha_t(s') \exp\left(\sum_{k} \lambda_k f_k(s', s, o, t)\right)$$

The backward procedure and Baum-Welch have been similarly modified. Z_o is given by $\sum_s \alpha_T(s)$. Viterbi algorithm for finding the most likely state sequence given the observation sequence have been modified from its HMM form.

CRFs have been shown to perform well in a number of natural language processing applications, such as POS tagging, shallow parsing or NP chunking, named entity recognition. Many participants of SemEval-2012 and i2b2 shared task used CRF for Temporal Information Extraction.

In the next section we will discuss features used for various Machine Learning approaches.

2.2.5 Feature Engineering

Feature engineering is a foremost essential task of Temporal Information Extraction for all classifiers. In this section, we describe various features that have been used in existing Time Expression Recognition systems. Features are descriptors or characteristic attributes of words designed for algorithmic consumption. Features can be specified in numerous ways using boolean values, numeric or nominal values. For example, a hypothetical Time Expression Detection system may be represented using 2 attribute:

- A Boolean attribute with the value true if word contains only numeric characters and false otherwise
- A numeric attribute corresponding to the length, in characters, of the word
- A nominal attribute corresponding to the lowercased version of the word

Features Examples	Examples
Case	Word is name of day or month Words is all numeric The word is mixed case (*e.g.*, April, Monday)
Punctuation	Ends with period, has eternal period (*e.g.*, a.m., p.m.)
Digit	Digit pattern Cardinal and ordinal Roman number Word with digits
Morphology	Prefix, suffix, singular version, stem Common ending
Part-of-speech	Cardinal (CD)
Function	Alpha, non-alpha, n-gram Lowercase, uppercase version Pattern, summarized pattern Token length, phrase length

Table 2.1: Word level features for Time Expression Detection

With above of set of features, the sentence "India got independence on 15th August 1947 " . This sentence can be represented using following feature vectors:

```
<false,5,"india">,<false,3,"got">,<false,12,"independence">,<false,2,"on">,
<false,4,"15th">,<false,6,"august">,<true,4,"1947">
```

Next, we describe the features that are most often used for the identification of time expressions. We organize the them in 2 categories: Word-level features and List lookup features.

Word features

Word level features are related to the character level feature of words. They specifically describe word case, punctuation, numerical value and special characters. Table 2.1 lists subcategories of word-level features.

Digit pattern

Digits can express wide range of useful information such as dates, percentages, intervals, identifiers *etc.* Certain patterns of digits gives strong signal about the type of Time Expression. For example, two digits and four digit numbers can stand for years and when followed by an "s", they can stand for a decade. Digits followed by am or pm stands for Time such as 10 am.

Functions over words

Features can be extracted by applying functions over words. An example a feature can be created by applying a non alpha function over the word to create word level features like $nonAlpha(I.B.M.) = ...$ Another method is to use character n-grams as features.

Patterns and summarized patterns

Pattern feature is to map words onto a small set patterns over character types. For instance, a pattern feature might map all uppercase letters to "A", all lower case letters to "a", all digits to "0" and punctuation to "-":

x="I.B.M": getPattern(x)="A-A-A-"

x="Model-123": getPattern(x)="Aaaaa-000"

The summarized pattern features is a condensed form of the above pattern feature in which consecutive similar pattern types that are repeated are removed. For instance, for the preceding example:

x="I.B.M": getPattern(x)="A-A-A-"

x="Model-123": getPattern(x)="Aa-0"

Unicode Character Encoding

Unicode character categories for each character of the token, with repeats merged (*e.g.* Feb29 would be 'LuLlNd')

Parse Tree Features

These features make use of dependency and/or constituency parse of the sentence. Features include governing verb, governing verb POS, governing preposition,phrase tag, path to root of parse tree, head word, head word lemma, head word POS.

Domain Features

Closed domain Temporal Information Extraction make use domain dependent features. Such features are derived by analysis of the corpora. For example, in medical discharge summaries, presence of words/tokens like "post-operative", "POD". Acronyms like "b.i.d" "t.i.d", "q.i.d" signals about presence of Time Expression. A Boolean attribute denoting the presence of such words have been proven to be beneficial for such task.

Window Features

Surrounding tokens helps in disambiguating the meaning of the word and hence make decision more deterministic. Context windows of 0, 1, 3, and 7 words preceding and following the token to be labeled are used. For Example, “May”followed by number is more likely to be time expression than being followed by an non numerical token.

Chapter 3

Approaches to Distributed Representation

In this chapter, we explore distributed word representation models. In cognitive science, central problem is to understand how agents represent information that enables them to behave in sophisticated ways. One big contention is whether the representation is localized or distributed. Contention remains whether knowledge is stored in specific, discrete region of brain or entire cortex. But with advent of connectionist models in mathematics, distributed representation has found great attention. Major benefit of using distributed representation is sharing of features to represent instance a knowledge. In most basic sense, a distributed representation is one that is spread out over a set of features for representation as opposed to localized approach where each feature is independent of each other. We will use distributed representation of words in our Neural network model for Temporal expression recognition. In next section, we see distributed representation for words in detail.

3.1 Distributed representation for words

A word representation is a mathematical object associated the each word, often a vector. Each dimension of the vector represents a feature and might even have a mathematical interpretation. Value of each dimension represents the amount of activity for that particular feature.

In machine learning, one of the most obvious model of representing a word is *one-hot vector representation*. In this representation only one of the computing element is active for each entity element. For example, if the size of vocabulary is $|V|$ then word w can be represented as vector of size $|V|$ in which the index of word w is only active and rest are set to zero.

```
Home : [0,0,0,0,....,1,....,0,0]
House: [0,0,0,0,..,1,..,0,0,0,0]
```

This representation is known as local representation. It is easy to understand and implement on

hardware. But this representation has many flaws of itself. As in example shown above, if we want the correlation between *Home* and *House*, the representation fails to show any correlation between the terms.

Lets take an example of POS tagging. We have

Training: "Dog slept on the mat"

Testing: "Cat slept on the mat"

By using localized vector representation, these two sentence would have completely different representation. Hence, a algorithm which has seen only *"Dog"* during training would fail to tag *"Cat"* during testing.

Distributed representation would represent these words in some lower dimensional dense vector of real values with each dimension representing a latent feature for word model. Distributed representation could be like :

```
Home : [0.112,0.432,.......,0.341]
House: [0.109,0.459,.......,0.303]
```

Distributed representation helps to solve the problem of sparsity. For words that are rare in the labeled training corpus, parameters estimated through one-hot representation will be poor. More over, the model cannot handle the word that do not appear in the corpus. Distributed representation are trained using large unlabeled corpus using an unsupervised algorithm. Hope is that the distributed representation would capture semantic and syntactic properties of word and would have a similar representation for syntactically and semantically related words.

For example, in the above example of POS tagging, even when we haven't seen *Cat* during training, distributed representation of *Cat* would be similar to *Dog*. Algorithm will be able to classify *Cat* with similar tag as it would have learned for *Dog*.

3.2 Training distributed word representation

Plethora of methods exists for dimensionality reduction and word representation. Usually, researcher use clustering for dimensionality reduction. There are mainly two types of clustering algorithms:

- **Hard clustering** : Class based model learn word classes based on distributional information. Words are then represented according to the representative of the class. Examples of hard clustering can be Brown clustering, Exchange clustering *etc.*
- **Soft clustering** : Soft clustering models learn for each cluster/topic a distribution over the words of how likely that word is in each cluster. Examples of soft clustering model are Latent Semantic Analysis (LSA/LSI), Latent Dirichlet Analysis (LDA), HMM clustering *etc.*

A continuous space word vector space representation differ significantly from traditional clustering methods. Words are represented using high dimensional dense vectors in continuous space with no

boundaries. In this study, we mainly focus on vector-space word representation that are learned by input layer of a neural networks. We will see three different methods based on neural network to learn vector representation from a large unannotated corpus.

3.2.1 Neural Probabilistic Language Model

Neural Probabilistic Language Model were introduced by Bengio et al. (2003). Method proposed by Bengio et al. is one of the first methods that introduces word vector representation to capture semantic similarity between the words. Primary aim of the paper is to develop a language model that overcomes the curse of dimensionality. For language model to be robust, it needs to generalize over the training instance. In higher dimensions, it is important how the algorithm distributes it probabilities mass around the training points. A algorithm performs better if probability mass is distributed where it matter rather than distributing it in all dimensions uniformly. Neural probabilistic model helps to achieve such important properties.

In short, the proposed approach can be summarized as follow :

1. associate each word with a distributed word feature vector
2. model the joint probability of word sequences in terms of the feature vectors of the words in sequence
3. learn word vectors and parameters of joint probability function simultaneously

The objective is to learn a good model $f(w_t, \cdots, w_{t-n+1}) = \hat{P}(w_t|w_1^{t-1})$ that give high out-of-sample likelihood. The function $f(w_t, \cdots, w_{t-n+1}) = \hat{P}(w_t|w_1^{t-1})$ is decomposed into two parts

1. A mapping C from any element from vocabulary V to a real vector $C(i) \in \mathbb{R}^m$. This is the distributed feature vector associated with every word in vocabulary.
2. The probability function over words, expressed with C: a function g maps an input sequence of feature vectors for words in context,$(C(w_{t-n+1}), \cdots, C(w_t))$, to a conditional probability distribution over words in V for the next word w_t

The output of function g is a vector whose i-th element estimates the probability $\hat{P}(w_t = i|w_1^{t-1})$ as shown in Figure-3.1. The function g is implemented either by feed-forward neural network or recurrent neural network or another parametrized function, with parameters ω. So overall parameter set is $\theta = (C, \omega)$. Training is achieved by looking for θ that maximizes the training corpus regularized log-likelihood:

$$L = \frac{1}{T} \sum \log f(w_t, w_{t-1}, \cdots, w_{t-n+1}; \theta) + R(\theta)$$

where $R(\theta)$ is regularization term. The network is trained in forward and backward pass. In forward pass the network computers the total log-likelihood. In backward pass the gradients are back propagated from the output layer layer to the input layer. The errors are also propagated to the vector mapping C as follow:

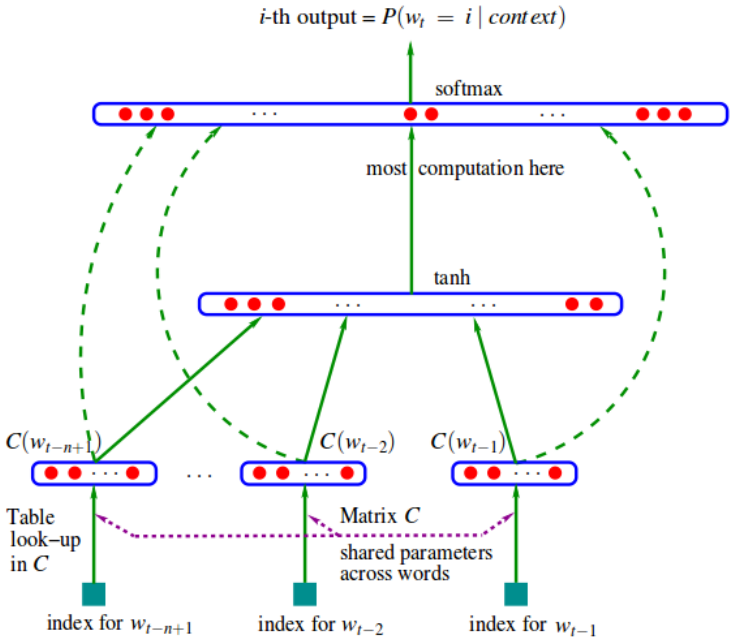

Figure 3.1: Neural architecture for language modeling

loop k between 1 to n-1

$C(w_{t-k}) \leftarrow C(w_{t-k})) + \epsilon \frac{\delta L}{\delta x(k)}$

end loop

where $\frac{\delta L}{\delta x(k)}$ is the the k-th block of the vector $\frac{\delta L}{\delta x}$

In this work, word vectors have been in terms of improvement in the measure of test set perplexity (geometric average of $\frac{1}{\hat{P}(w_t|w_1^{t-1})}$. Experiment shows 24% improvement in perplexity on Brown corpus in comparison to n-gram technique.

3.2.2 Contrastive criterion learning

Collobert and Weston (2008) in their paper. *A Unified Architecture for Natural Language Processing: Deep Neural Network for Multitask Learning*, proposed a language model that can utilize large unlabeled corpus for learning a distributed representation. Motivation of the paper is to develop a unified architecture that can perform various NLP task like POS tagging, Chunking, Named Entity Recognition and Semantic role labeling.

All of these tasks can be seen as tasks of labeling words or chunks. Against the traditional NLP, wherein features are handcrafted and fed to a classical shallow classification algorithm like Support vector machine (SVM), algorithm should learn the necessary features automatically. Choice of feature is quite empirical and mainly based on trial and error. A better approach would be in which machine learns a good set of features on itself. A deep neural network is used in this study. Features for the network are automatically trained using backpropagation algorithm. Figure-3.2 summarizes the architecture of the

system.

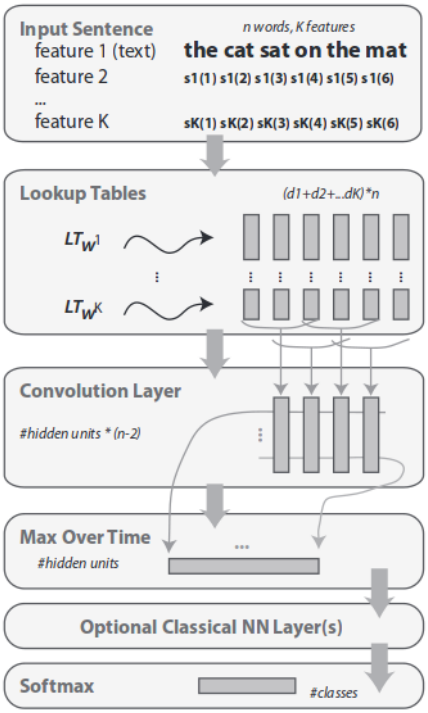

Figure 3.2: Multitask learning architecture

Foundation of the multitask learning lies in learning very good word representation from which higher level feature could be extracted for the specific needs of a task. In Collobert et al. (2011), detailed explanation of unsupervised trainer is presented first introduced in Collobert and Weston (2008). Collobert et al. proposes a pairwise ranking approach based on Cohen et al. (1999). Motivation of ranking pair of phrases is to score a legal phrase higher than an incorrect phrase. Figure-3.3 shows the network design. Architecture considers a window approach network with parameters θ which outputs a score $f_\theta(x)$ given a window of text $x = [w]_1^{d_w in}$. As in Figure-3.3, network consists of one non-linear hidden layer and a scoring node as the output.

$$
\begin{aligned}
a &= s(W^T x + b) \\
f_\theta(x) &= U^T a \\
a_c &= s(W^T x_c + b) \\
f_\theta(x_c) &= U^T a_c \\
\theta &\rightarrow \sum_{x \in \Phi} \sum_{w \in D} \max\left(0, 1 - f_\theta(x) + f_\theta(x_c)\right)
\end{aligned}
$$

Network is first shown a positive sample and then a corrupted sample. Scores $f_\theta(x)$ and $f_\theta(x_c)$ are obtained. If the sample don't separate substantially, then the error is propagated back in the network to adjust the parameters of the network. In next approach, we will see the model that improve the runtime

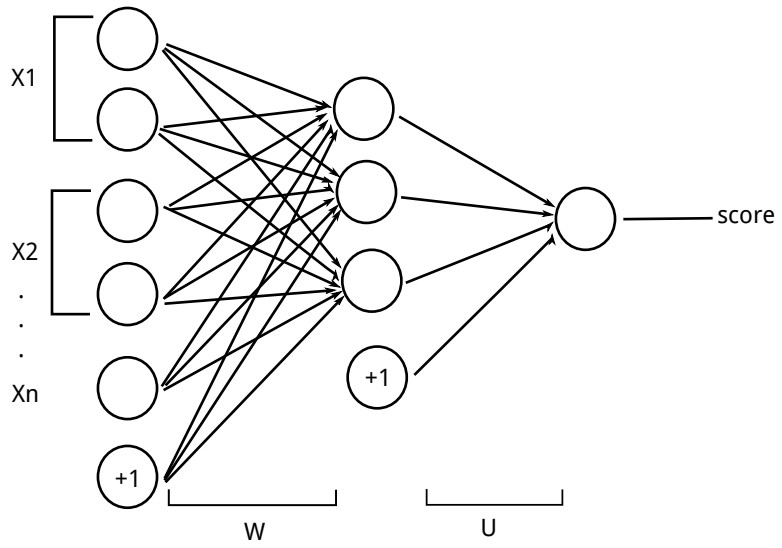

Figure 3.3: Network architecture for discriminative learning

of the architecture.

3.2.3 Combined Bag of Words Approach (CBOW)

In the previous model, we see that the architecture proposed generally has four layer. At the input layer, N previous words are encoded using 1 of V coding, where V is the size of vocabulary. The input layer is then projected to projection layer P of dimensionality NxD, using shared projection matrix. This step was implicit in the method of concatenating feature vectors from a lookup table. Model becomes complex for computation between projection and hidden layer as values are real and dense. For a window size $N = 5$, projection layer could be of size 250 to 1000, while the hidden layer size could be typically 250 to 750 units. Moreover, hidden layer compute probability distribution over all the words in the vocabulary expect in contrastive estimation method. Thus the complexity per each training example is

$$Q = N \times D + N \times D \times H + H \times V$$

In above equation $H \times V$ is the dominating term because of non-linear nature of the layer. To avoid the complexity of hidden layer , a new log linear model is proposed to avoid non-linear hidden layer but able to represent data as precisely as neural networks. Continuous bag of word architecture removes the hidden layer abd the projection layer is shared for all words (not just the projection matrix). All words from a window gets projected into the same position (their vectors are averaged). Since the order of words is not important, model is called bag-of-word model. A log linear classifier is used to classify the current word given the window w along the past and future of the word under consideration. Weight matrix between the input and projection layer is shared for all inputs. A further optimization is obtained

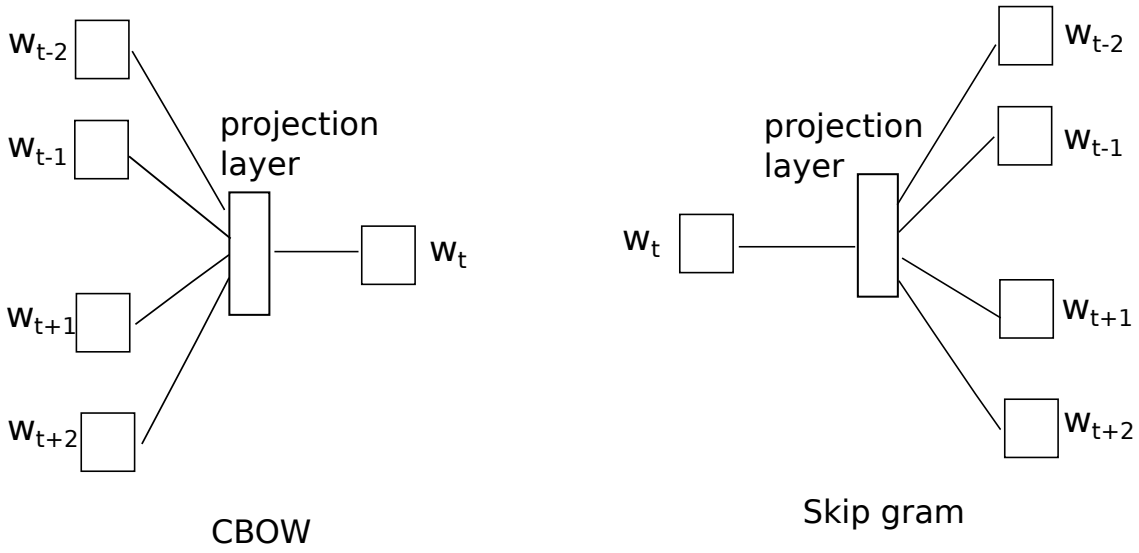

Figure 3.4: CBOW and Skip gram approach

using hierarchical softmax. Complexity can be expressed as:

$$Q = N \times D + D \times \log_2(V)$$

3.2.4 Continuous Skip-gram approach

In the same paper, Mikolov et al. (2013) proposes second efficient model to generate word representation. Architecture is similar to CBOW but instead of predicting the current word based on context, method tries to maximize classification of a word based on other words in a sentence. Method uses current word as an input to the projection layer and try to predict the words within certain range before and after the current word.

Figure 3.4 shows the architecture of the model. The complexity of the model can be expressed as :

$$Q = C \times (D + D \times \log_2(V))$$

here C is maximum distance from which we want to predict the word. We random choose R between 1 and C and then we use R words from past and R words from the future. Since the words are randomly chosen we skip some of the words in context and hence the name skip-gram is used for model.

3.3 Semantic and syntactic information in representation

Continuous word representation capture lot of syntactic and syntactic similarities of words in their dense compact representation. Many linear dependencies among the words are captured using the model

discussed in previous section. As we will show in result section of the report, semantic and syntactic similarities between the words are well captured by the model. Representation for all the states were similar in the vector space. All person names , city names were distinctly represented in the space. Syntactic properties like many inflectional form of word ,*viz.* dukan and dukhano ,were nearest neighbor of each other.

Surprisingly, the vectors model has very nice vector properties. We can answer some analogy question use simple algebraic operations with the vector representation of the words. For example to find a word, that is similar to *small* in the same sense as *biggest* is similar to *big*, we can simply compute $X = vector(\text{"}biggest\text{"}) - vector(\text{"}big\text{"}) + vector(\text{"}small\text{"})$. If we find words with similar representation as X using cosine similarity and use it to answer the query then it is possible that one of the option would be "smallest" among the possibilities.

If large training corpus and big vectors are available and model is trained on them, it is expected that more semantic and syntactic information could be captured using word representation model. We may be able to respond to semantic query like analogy among date expressions. For example , Washington is to USA as Delhi is to India. Word vectors with such semantic properties has lot of potential application in machine translation, information retrieval, question answering systems and many other applications.

Chapter 4

Approches to Event Extraction

Approaches to event extraction are broadly classified into three categories based on modeling techniques. Firstly, there are data-driven approaches as discussed in Section 4.1 which exploits statistics, machine learning and linear algebra to convert data into knowledge. Secondly, there exists knowledge-driven as discussed in Section 4.2 methods which uses expert knowledge encoded in the form of patterns or rules to extract knowledge. Finally, hybrid event extraction approaches as discussed in Section 4.3 combines data-driven and knowledge based methods.

4.1 Data-Driven Event Extraction

Data-driven approaches are commonly used for natural language processing applications. These approaches rely solely on quantitative methods to discover relations. Data-driven approaches require large text corpora in order to develop models that approximate linguistic phenomena.

STEP (Bethard and Martin, 2006) is a system for TimeML event recognition and classification. This approach uses a rich set of textual, morphological, dependency, temporal and WordNet hypernymy features to build a Support Vector Machine (SVM) model.

March and Baldwin (2008) present an evaluation on event recognition using a multi-class classifier (BSVM). The main features used to train the classifier are word, PoS context window, stop words removal and feature generalization through words grouping (numbers, named entities, *etc.*). The result for the best feature combination over TimeBank was 76.4% F-Score. The novelty in their work is making use of feature space reduction technique by removing stop words and combining Numbers and Named Entities.

TempEval-2013 (Temporal Evaluation) which was taken as a shared task in SemEval workshop observed many submission exploiting data driven approaches.

Lei et al. (2005) present a novel approach to detect and track of events in news articles using subject extraction and improved Support Vector Machines. , in which subject concepts can concisely and precisely express the meaning of a longer text. These concepts are extracted from news article headlines. According to authors headlines carries information about who, what and why. Then improved SVM first prunes

the negative examples, reserves and deletes a negative sample according to distance and class label, then trains the new set with S VM to obtain a classifier and maps the SVM outputs into probabilities.

Okamoto and Kikuchi (2009) employs hierarchical clustering techniques for detecting volatile events in neighborhood. Two level clustering algorithm was proposed. Firstly, blog entries using geographic names are searched,collected and stored in repository. Second, document vectors are generated from the collected entries using morphological analysis, named entity recognition, and IDF-based weighting function. Next, similar documents are clustered together using cosine measure. This method produces clusters such that each cluster corresponds to a different topic. Finally, A topic word is extracted from each topic cluster with a modified version of the C-value method.

A **drawback** of the discussed data-driven methods to event extraction is that they do not deal with meaning explicitly, *i.e.* they discover relations in corpora without considering semantics. Another disadvantage of statistics-based text mining is that a large amount of data is required in order to get statistically significant results. However, since these approaches are not based on knowledge, neither linguistic resources, nor expert (domain) knowledge are required.

4.2 Knowledge-Driven Event Extraction

In contrast to data-driven methods, knowledge-driven methods exploits human knowledge about the contents of the text to be processed in addition to linguistic and lexicographic knowledge . This knowledge is encoded in the form of patterns that express rules which are used to extract desired information from the text. Information is mined from corpora by using predefined or discovered linguistic patterns, which can be either lexico-syntactic patterns or lexico-semantic patterns. While Lexico-syntactic exploits lexical representation with syntactical information, lexico-semantic uses semantic or meaning of the information along with lexical representations.

EVITA (Saurí et al., 2005) is an application for recognizing events in natural language texts for Question Answering System. It recognizes events by applying linguistic rules encoded in the form of patterns. It considers Verb Phrases (NP), Noun Phrases(NP) and Adjectival Phrases(ADJP) as most probable candidates for containing events. It employs different strategies for extracting events from Verbal, Noun and Adjectival Chunks. It make use Part-of-speech (PoS) tagging, lemmatizing, chunking, lexical lookup and contextual parsing to manually encode event extraction rules. Furthermore, WordNet information combined with Bayesian learned disambiguation was used to identify nominal events.

Piskorski et al. (2007) extracted Violent Events and security related facts from on-line news. It uses a pipeline where news articles which are collected by scanning news headlines. Collected news article are clustered together through document matching technique , such that each cluster corresponds to one topic. For each document in a cluster text processing techniques which are linguistically motivated are applied to extract main events. In this phase geo-coding is also performed to identify the place where main event took place. Pattern matching techniques are applied to news text document to capture

the facts about the events. Finally events extracted from individual documents are clustered to give cross-document clustered view of events. For this task simple voting-like heuristics are deployed.

REES (Aone and Ramos-Santacruz, 2000) is a large scale event and relation extraction system. Here, lexico-syntactic patterns are applied so as to discover a wide range of relations and events. REES consists of three components: Tagger, Co-reference resolver and a Template generator. Tagger module exploits pattern matching rules to extract event using lexicon-driven, syntactically-based generic patterns. The important aspect of REES is its declarative, lexicon-driven approach. This approach requires a lexicon entry for each event-denoting word, which is generally a verb. The lexicon entry specifies the syntactic and semantic restrictions on the verb's arguments. Co-reference resolution module resolves only definite noun phrases of Organization, Person, and Location types, and singular person pronouns: he and she.

Hung et al. (2010) presented a method for extracting event-based commonsense knowledge by using lexico-syntactic pattern matching and semantic role labeling. Authors first extract raw sentences from web based on lexico-syntactic pattern matching. Patterns like *subject + "is capable of"+ verb*, *subject + "is able to "+ verb* and many others are used to extract sentences from web. Each sentence is parsed through a Semantic Role labeling module to extract verbs and its arguments. Verbs along with its arguments forms a knowledge but this method is prone to errors so, authors proposed a strategy for semantic role plausibility verification, based on a semantic role substitution strategy, which significantly pruned knowledge items with a high probability of erroneously parsed semantic roles.

Nishihara et al. (2009) extracted events from personal blog experiences. Triplet of place, object and action is considered as events. Authors employs simple linguistic rules to extract triplet information. Nouns following a preposition like *at, in, on* is regarded as place. Noun that co-occur with noun keyword denoting place is taken as an object and verbs occurring in a sentences which contains object nouns are taken as actions.

Knowledge based approach to event extraction was also used by Sprugnoli and Lenci in TempEval 2013 task.

To summarize Pattern-based approaches has an advantage of using very less amount of data. It exploits syntactic and semantic pattern encoded in the form of rules to extract desired information. However, feasibility and accuracy of rules depends on users knowledge of linguistics as well as domain expertise.

4.3 Hybrid Event Extraction

Hybrid information extraction techniques exploits advantages of Data-driven and knowledge based approaches. Authors have applied expert knowledge to the output of statistical model to prune unwanted results or to include information that could have been missed by statistical models. Also, researchers have combined statistical approaches with (lexical) knowledge, where lexical knowledge is encoded as features for statistical learning.

Kolya et al. (2011) combined statistical model with semantic and lexical knowledge to extract event and event actors. Initially, SVM classifier was trained on TempEval dataset to extract event words or phrases. As per authors observations deverbal event nouns are not correctly identified. In order to solve this problem authors introduce several strategies in conjunction with machine learning. These strategies were based on handcrafted rules exploiting Semantic role-labeling and WordNet. Semantic roles was used to detect the events that are the nominalizations of verbs such as agreement for agree or construction for construct. Verbal event nouns like *war, tour* which are also used as verbs were not being identified as events. Authors used WorndNet Noun and Verb Hierarchy to identify such event words. Lastly, linguistic rules were used in-order to identify words with suffixes such as *-cion, -tion or -ion.*

Chinese News Fact Extractor (Wang et al., 2010) aimed at extracting 5W1H (Who, What, Whom, When, Where and How) semantic elements from Online News. Authors proposed a novel algorithm to extract topic sentences by stressing the importance of news headline; Then event facts (*i.e.* 5W1H) were extracted from these topic sentences by applying a rule-based method (verb-driven) and a supervised machine-learning method (SVM). Firstly topic sentences are extracted by calculating overlap between words in headlines and news text. Next, SVM classifier is trained with morphological features like POS Tag, Sentence length, word position to extract candidate event words. Finally rules based on valence grammar and previous two stages are used to find 5W1H elements. Authors used the notion of Valency Grammar to construct syntactic rules to extract verbs and its arguments. Univalent Verbs or intransitive Verbs follows syntactic construction of NP+V or V+NP, bivalent transitive verbs follows construction of NPl + V + NP2, NP1 + PNP2 + V and lastly trivalent transitive verbs follows construction of NP1 + V + NP2 + NP3, NP1 + PNP2 + V + NP3 where NPare Noun Phrases, V is Verb, PNPProper Noun Phrases.

In hybrid event extraction systems, due to the usage of data-driven methods, the amount of required data increases, yet typically remains less than is the case with purely data-driven methods. Compared to a knowledge-driven approach, complexity – and hence required expertise – increases due to the combination of multiple techniques. On the other hand, the amount of expert knowledge that is needed for effective and efficient event discovery is generally less than for pattern-based methods, because of the fact that lack of domain knowledge can be compensated by the use of statistical methods.

4.4 TempEval 2013 Task of Event Extraction

The Semantic Evaluation (SemEval) workshop focus on the evaluation of semantic analysis systems. In 2013, TempEval-3 which was taken as a shared task in SemEval 2013 workshop. It aims on doing advance research on temporal information processing, which could eventually help NLP applications like question answering, textual entailment, summarization, *etc.* TempEval Task B, **Event Extraction and Classification** was aimed to determine the extent of the events in a text as defined by the TimeML EVENT tag and their appropriate CLASS. In this section we will describe participants' approaches,

results, and the observations from their approaches and results.

4.4.1 Participants and their approaches

For Event Extraction task, 12 submissions were made from 7 participants. Data-driven and Rule-based approaches were exploited. As discussed in section 4.1 data-driven approaches require large text corpora in order to develop models that approximate linguistic phenomena. It also relies on statistical reasoning based on probabilistic modeling, information theory, and linear algebra. Figure 4.1 shows the participants those exploited data-driven approaches, dataset which was used for training , and classifiers used by the participants.

ATT (Jung and Stent, 2013) have explored the trade-off between additional context on the one hand, and additional layers of representation on the other. Authors have investigated the impacts of different sets of features and also examined performance based on different sizes of n-grams in a small scale (n=1,3). Authors explored from simplistic features like POS Tag, token, lemma to more complicated representations like Semantic role labels. They also experimented with context windows of 0, 1, 3, and 7 words preceding and following the token to be labeled (*i.e.* window sizes of 1, 3, 7, and 15).

Strategy	System	Training data	Classifier used	Linguistic Knowledge
Data-driven	ATT-1, 2, 3	TBAQ + TE3Silver	MaxEnt	*ms, ss*
	ClearTK-1, 2	TimeBank	SVM, Logit	*ms*
	ClearTK-3, 4	TBAQ	SVM, Logit	*ms*
	JU-CSE	TBAQ	CRF	
	KUL	TBAQ +TE3Silver	Logit	*ms, ls*
	KUL-TE3RunABC	TBAQ +TE3Silver	Logit	*ms, ls*
	NavyTime-1	TBAQ	MaxEnt	*ms, ls*
	NavyTime-2	TimeBank	MaxEnt	*ms, ls*
	Temp : ESAfeature	TBAQ	MaxEnt	*ms, ls, ss*
	Temp : WordNetfeature	TBAQ	MaxEnt	*ms, ls*
	TIPSem (TE2)	TBAQ	CRF/SVM	*ms, ls, ss*
Rule-based	FSS-TimEx (EN)	None	None	*ls, ms*
	FSS-TimEx (ES)	None	None	*ls, ms*

Figure 4.1: Data-driven approaches for Event Extraction

ClearTK (Bethard, 2013) had focused on using small set of simple features that can be derived from either tokens, part-of-speech tags or syntactic constituency parses. Authors of the system were primarily interested in evaluating how useful the different corpora are. They submitted 4 system numbered ClearTK - 1,2,3,4 with different size of training and testing data. From results it can be inferred that minimalistic set of features was beneficial for tasks like relation detection between entities, it could not achieve good results event identification and classification task.

JU-CSE (Kolya et al., 2013) had explored the combinations of Wordnet hypernym, hyponym, and

other semantic and lexical relations in Wordnet and semantic role labels to train CRF classifier. Inspite of using rich semantic features system could not achieve good performance results.

NavyTime (Chambers, 2013) system also made use of minimalistic set of features derived from tokens, part-of-speech tags, constituency and dependency parse trees to train Maxent classifier. Authors of the system experimented by having separate classifier for inter-sentence event-event pairs and another for intra-sentence event-event pairs. System performed well for event identification task but could not achieve high score in event classification.

KUL (Kolomiyets and Moens, 2013) used multi-label logistic regression classifier for event detection and classification with features derived from dependency and constituency parse trees and shallow parsing.

Authors of the system Temp:ESAFeature have experimented with Explicit Semantic Analysis scores and Wordnet Hypernym as features for classifying the event and types.

To summarize, researchers have experimented with different combination of morphological, lexical and semantic features. A very common observation is the use of POS Tags,Word Lemma, Word Stem as a feature for ML classifiers. These features have been proven to be very beneficial for various NLP tasks. Use of features like semantic roles, ESA, Wordnet lexical and semantic relations are proven to be beneficial for event identification but could not make significant contribution to event classification task. Also, choice of classifier plays a crucial role in systems performance. Treating problem of classification as a multi-label classification task or sequence labeling task impacts performance values.

Knowledge-driven text mining, as discussed in section 4.2 is often based on patterns that express rules representing expert knowledge. It is inherently based on linguistic and lexicographic knowledge, as well as existing human knowledge regarding the contents of the text that is to be processed. It may require no or very minimal amount of data for extracting knowledge from text but makes heavy use of linguistic phenomenon. Figure 4.2 shows the participants those who have used rule based methods for the task of identification and classification of events.

Strategy	System	Training data	Classifier used	Linguistic Knowledge
Rule-based	FSS-TimEx (EN)	None	None	*ls, ms*
	FSS-TimEx (ES)	None	None	*ls, ms*

Figure 4.2: Rule Based approaches for Event Extraction

Single submission was made by Sprugnoli and Lenci. FSS-TimEx was developed as part of a multilingual event extraction system, Nexus, which runs on top of the EMM news processing engine. The system was actually made to extract highly domain specific events (like Armed Conflict, Earthquake, Terrorist Attacks). To make it compliant to TempEval task, which aims to extract domain independent events authors made use of small set of language-dependent finite-state rules to model verb phrase structure. In order to determine the Class for the event extraction task, authors experimented with using a language-

independent method for weakly supervised lexical acquisition. The algorithm takes as input a small set of seed terms, an unannotated text corpus and a parameter for the number of bootstrapping iterations: it then learns a ranked list of further terms, which are likely to belong to the same class, based on distributional n-gram features and term clustering. But this approach could not achieve good score for Event classification. It ranked lowest amongst all participants.

4.4.2 Observations

	F1	P	R	**class F1**
ATT-1	**81.05**	81.44	80.67	**71.88**
ATT-2	80.91	81.02	**80.81**	71.10
KUL	79.32	80.69	77.99	70.17
ATT-3	78.63	**81.95**	75.57	69.55
KUL-TE3RunABC	77.11	77.58	76.64	68.74
ClearTK-3,4	78.81	81.40	76.38	67.87
NavyTime-1	80.30	80.73	79.87	67.48
ClearTK-1,2	77.34	81.86	73.29	65.44
NavyTime-2	79.37	80.52	78.26	64.81
Temp:ESAfeature	68.97	78.33	61.61	54.55
JU-CSE	78.62	80.85	76.51	52.69
Temp:WordNetfeature	63.90	78.90	53.69	50.00
FSS-TimEx	65.06	63.13	67.11	42.94
TIPSem (TE2)	82.89	83.51	82.28	75.59

Figure 4.3: Rule Based approaches for Event Extraction

Figure 4.3 shows the results of different approaches. Systems are ranked based on F1-score of class event attribute. All systems except one made use of machine learning approaches. Three datasets were used and it was observed that size of training data plays an important role in performance values. Systems evaluated on silver dataset (TE3) along with two gold standard dataset(Timebank and AQUAINT) performed better than systems evaluate only on gold standard datasets. As discussed earlier, morphosyntactic information, *e.g.* POS, lexical information, morphological information and syntactic parsing related features; lexical semantic information, *e.g.* WordNet synsets; and sentence level semantic information, *e.g.* Semantic Role labels were explored. Highest F1 measure for event classification task is 81.95% and that for event classification is 71.88%. We expect that more improvements can be made by better feature selections and this will the main focus of our research work.

Bibliography

David Ahn. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8. Association for Computational Linguistics, 2006.

Chinatsu Aone and Mila Ramos-Santacruz. Rees: a large-scale relation and event extraction system. In *Proceedings of the sixth conference on Applied natural language processing*, pages 76–83. Association for Computational Linguistics, 2000.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=944919.944966`.

Steven Bethard. Cleartk-timeml: A minimalist approach to tempeval 2013. In *Second Joint Conference on Lexical and Computational Semantics (* SEM)*, volume 2, pages 10–14, 2013.

Steven Bethard and James H Martin. Identification of event mentions and their semantic class. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 146–154. Association for Computational Linguistics, 2006.

Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. An algorithm that learns whats in a name. *Mach. Learn.*, 34(1-3):211–231, feb 1999. ISSN 0885-6125. doi: 10.1023/A:1007558221122. URL `http://dx.doi.org/10.1023/A:1007558221122`.

Nathanael Chambers. Navytime: Event and time ordering from raw text. Technical report, DTIC Document, 2013.

Angel X Chang and Christopher D Manning. Sutime: A library for recognizing and normalizing time expressions. In *LREC*, pages 3735–3740, 2012.

William W. Cohen, Robert E. Schapire, and Yoram Singer. Learning to order things. *J. Artif. Int. Res.*, 10(1):243–270, May 1999. ISSN 1076-9757. URL `http://dl.acm.org/citation.cfm?id=1622859.1622867`.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine*

learning, ICML '08, pages 160–167, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390177. URL http://doi.acm.org/10.1145/1390156.1390177.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November 2011. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1953048.2078186.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

Jr. Forney, G.D. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973. ISSN 0018-9219. doi: 10.1109/PROC.1973.9030.

Patrick Haffner. Scaling large margin classifiers for spoken language understanding. *Speech Communication*, 48(3):239–261, 2006.

Sheng-Hao Hung, Chia-Hung Lin, and Jen-Shin Hong. Web mining for event-based commonsense knowledge using lexico-syntactic pattern matching and semantic role labeling. *Expert Systems with Applications*, 37(1):341–347, 2010.

Prateek Jindal and Dan Roth. Extraction of events and temporal expressions from clinical narratives. *Journal of biomedical informatics*, 46:S13–S19, 2013.

Hyuckchul Jung and Amanda Stent. Att1: Temporal annotation using big windows and rich syntactic and semantic features. In *Second Joint Conference on Lexical and Computational Semantics (* SEM)*, volume 2, pages 20–24, 2013.

Oleksandr Kolomiyets and Marie-Francine Moens. Kul: A data-driven approach to temporal parsing of documents. In *Proceedings of the second joint conference on lexical and computational semantics (* SEM), Volume 2: Proceedings of the seventh international workshop on semantic evaluation (SemEval 2013)*, pages 83–87, 2013.

Anup Kumar Kolya, Asif Ekbal, and Sivaji Bandyopadhyay. A hybrid approach for event extraction and event actor identification. In *RANLP*, pages 592–597. Citeseer, 2011.

Anup Kumar Kolya, Amitava Kundu, Rajdeep Gupta, Asif Ekbal, and Sivaji Bandyopadhyay. Ju_cse: A crf based approach to annotation of temporal expres-sion, event and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (* SEM)*, volume 2, pages 64–72. Citeseer, 2013.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA,

2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-778-1. URL http://dl.acm.org/citation.cfm?id=645530.655813.

Zhen Lei, Ying Zhang, Yu-chi Liu, et al. A system for detecting and tracking internet news event. In *Advances in Multimedia Information Processing-PCM 2005*, pages 754–764. Springer, 2005.

Yu-Kai Lin, Hsinchun Chen, and Randall A Brown. Medtime: A temporal information extraction system for clinical narratives. *Journal of biomedical informatics*, 46:S20–S28, 2013.

Inderjeet Mani and George Wilson. Robust temporal processing of news. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 69–76. Association for Computational Linguistics, 2000.

Olivia March and Timothy Baldwin. Automatic event reference identification. In *Australasian Language Technology Association Workshop 2008*, volume 6, pages 79–87, 2008.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffery Dean. Efficient estimation of word representations in vector space. page to appear, 2013.

Matteo Negri and Luca Marseglia. Recognition and normalization of time expressions: Itc-irst at tern 2004. *Rapport interne, ITC-irst, Trento*, 2004.

Yoko Nishihara, Keita Sato, and Wataru Sunayama. Event extraction and visualization for obtaining personal experiences from blogs. In *Human Interface and the Management of Information. Information and Interaction*, pages 315–324. Springer, 2009.

Masayuki Okamoto and Masaaki Kikuchi. Discovering volatile events in your neighborhood: Local-area topic extraction from blog entries. In *Information Retrieval Technology*, pages 181–192. Springer, 2009.

Jakub Piskorski, Hristo Tanev, and Pinar Oezden Wennerberg. Extracting violent events from on-line news for ontology population. In *Business Information Systems*, pages 287–300. Springer, 2007.

James Pustejovsky, Bob Ingria, Roser Sauri, Jose Castano, Jessica Littman, Rob Gaizauskas, Andrea Setzer, Graham Katz, and Inderjeet Mani. The specification language timeml. *The language of time: A reader*, pages 545–557, 2005.

Kirk Roberts, Bryan Rink, and Sanda M Harabagiu. A flexible framework for recognizing events, temporal expressions, and temporal relations in clinical text. *Journal of the American Medical Informatics Association*, 20(5):867–875, 2013.

Roser Saurí, Robert Knippen, Marc Verhagen, and James Pustejovsky. Evita: a robust event recognizer for qa systems. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 700–707. Association for Computational Linguistics, 2005.

Sunghwan Sohn, Kavishwar B Wagholikar, Dingcheng Li, Siddhartha R Jonnalagadda, Cui Tao, Ravikumar Komandur Elayavilli, and Hongfang Liu. Comprehensive temporal information detection from clinical text: medical events, time, and tlink identification. *Journal of the American Medical Informatics Association*, pages amiajnl–2013, 2013.

Rachele Sprugnoli and Alessandro Lenci. Crowdsourcing for the identification of event nominals: an experiment.

Jannik Strötgen and Michael Gertz. Heideltime: High quality rule-based extraction and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 321–324. Association for Computational Linguistics, 2010.

Marc Verhagen and James Pustejovsky. Temporal processing with the tarsqi toolkit. In *22nd International Conference on on Computational Linguistics: Demonstration Papers*, pages 189–192. Association for Computational Linguistics, 2008.

Wei Wang, Dongyan Zhao, Lei Zou, Dong Wang, and Weiguo Zheng. Extracting 5w1h event semantic elements from chinese online news. In *Web-Age Information Management*, pages 644–655. Springer, 2010.

Yan Xu, Yining Wang, Tianren Liu, Junichi Tsujii, I Eric, and Chao Chang. An end-to-end system to identify temporal relation in discharge summaries: 2012 i2b2 challenge. *Journal of the American Medical Informatics Association*, 20(5):849–858, 2013.