Reconciling the output of multiple stochastic classifiers subject to global rules:
Detecting and Classifying Temporal Relations in Text

UCD DUBLIN
UCD Michael Smurfit Graduate Business School

Terri Hoare, B.Sc. Computer Science

Catherine Kerr, B.Sc. Computer Science

A thesis submitted to University College Dublin in part fulfilment of the requirements of the degree of M.Sc. in Business Analytics

Michael Smurfit Graduate School of Business

September, 2014

Supervisor(s): Dr. Paula Carroll, UCD

Dr. Jakub Mareček, IBM

Head of School: Professor Ciarán Ó hÓgartaigh

Dedication

This work is dedicated to …

"The loving memory of Stuart

As a tribute to my parents Jean and Joe

To a future for Kate and Meagan

where all things are possible"

- Terri

and to …

Michael

"Thanks for being the cook, the cleaner,

the shopper and the driver for the last year

- Catherine

Table of Contents

List of Figures

List of Tables

Preface

As a result of the very rapid growth and increasingly fast pace of online newsfeeds, there are many applications for an automated temporal ordering of newsfeed events. There has been much recent progress in being able to temporally relate events in newsfeeds, notably the adoption of the ISO TimeML standard in 2005. The TempEval shared task was established in order to advance research in machine learning on temporally labelled newsfeed documents. The recent TempEval challenge focused on the detection and classification of the relationship types in a newsfeed and thirteen classifiers participated. The aim of this dissertation is to combine the output from these classifiers into an ensemble classifier that is more accurate than its constituents and to apply Integer Programming techniques to reconcile the classifiers according to global rules.

Dublin, Terri Hoare
September 2014 Catherine Kerr

Acknowledgements

We would very much like to thank Dr. Jakub Mareček of IBM Research for proposing this dissertation and for giving us such support throughout. It was a wonderful journey and we could not have achieved our objectives without his help and guidance. We also thank Dr. Paula Carroll, our academic supervisor, for her continued support, advice and enthusiasm and for always keeping the door open.

Special thanks also to the participants of the TempEval-3 challenge who so kindly shared their data and ideas with us - Dr. Nate Chambers (Navytime), Dr. Natsuda Laokulrat (UTTime) and Dr. Steven Bethard (ClearTK). Also to Dr. Marc Verhagen and Dr. Naushad UzZaman of the TempEval-3 team.

And of course, we would also like to thank our families for making the journey that much easier.

Abstract

Applications for automated temporal reasoning between events in newsfeed documents include situational reporting to assist in emergency relief efforts as well as national security intelligence alerts. State of the art machine learning techniques have been applied to build classifiers trained on large hand-annotated corpora to predict relationships between events in a newsfeed document. A weakness of machine learning is that it does not perform global consistency checking. We aim to build an experimental model to test whether a statistical ILP ensemble can improve on the accuracy and quality of the relation predictions. Based on data from the 2013 TempEval Challenge and using their quality metric, we are able to show an improvement of three percentage points in temporal awareness score, equivalent to an 8% improvement on the current best classifier. This improvement was achieved running on open source software on our laptops. We believe that this estimate can easily be improved with an increase in the number and diversity of component classifiers; a commercial solver; increased computer power, and increased training data for the estimation of a stronger Bayesian prior distribution required for a full application of a Bayesian statistical ensemble. An accurate and temporally consistently labelled newsfeed document can be used for automated inferencing using induced timelines on the relations.

> "*The actual science of logic is conversant at present only with things either certain, impossible, or entirely doubtful…therefore the true logic for the world is the calculus of probabilities, which takes account of the magnitude of probability which is, or ought to be, in a reasonable man's mind*"
>
> – James Clerk Maxwell.

List of important abbreviations

(compiled using Mathematica and Wikipedia)

AI	Artificial Intelligence
Cbc	Coin-or branch and cut
Coin-OR	The Computational Infrastructure for Operations Research project.
DARPA	Defence Advanced Research Projects Agency
ICRC	International Committee of the Red Cross
ILP	Integer Linear Programming
IP	Integer Programming
ISO	International Organization for Standardization
LP	Linear Programming
LRCS	League of Red Cross Societies
NLP	Natural Language Processing
OR	Operations Research
Sitreps	Situational Reports
SRL	Semantic Role Labeling
UNDRO	United Nations Disaster Relief Organisation
UNHCR	United Nations High Commissioner for Refugees

Chapter 1 - Introduction

1.1 Background

Since the pioneering work of Granger (Nobel Prize, 2003) (Granger and Newbold 1977), there has been much research into the combination of forecasts and classifiers.

In machine learning, **ensemble methods** are used to obtain accurate classifiers by combining less accurate ones, intuitively voting amongst the predictions for the most likely prediction. Unlike a statistical ensemble which builds a probability distribution for the state of the system from a sometimes infinite set of possible copies of the system, a machine learning ensemble refers only to a concrete finite set of alternative models, but typically allows for a much more flexible structure to exist between those alternatives. It has been established that a necessary and sufficient condition for an ensemble of classifiers to be more accurate than any individual members is if there is sufficient accuracy and diversity in prediction amongst the constituent classifiers (Dietterich 2000).

Current research in Natural Language Processing (NLP), the ability for computers to derive meaning from human natural language, is based on statistical machine learning techniques. Within this broader context, our research focuses on the ability of computers to detect the temporal relationships between the events in newsfeed documents. The complexity of temporally labelling the relations in a newsfeed document is on a par with the other challenging semantic labelling tasks in NLP. Temporal relationship labelling includes first identifying related pairs of temporally labelled events or times and then classifying the ordering relationship between each pair. Human inter-annotator agreement is typically only fifty-five percent due to the large number of pairs that can be selected for comparison (Mani et al. 2006)

As a result of the very rapid growth and increasingly fast pace of online newsfeeds, there are many applications for an automated temporal ordering of newsfeed events. For example a Situational Report (Sitrep) is *"a brief report that is published and updated periodically during a relief effort and which outlines the details of the emergency, the needs generated and the responses undertaken by all donors as they* *become known. Sitreps are issued by UNDRO, by UNHCR, ICRC and LRCS."* [1] During the recent (2014) floods in the Balkan countries, news reports provided up-to-date information on the extent of the disaster, including tallies on the numbers dead, injured, or displaced. Figure 1-1 shows a series of headlines on Reuters.com.

May 15 2014: Five dead as worst floods in 120 years hit Serbia, Bosnia

May 18 2014: Balkans flooding threatens Serbia power plants, 37 dead

May 19 2014: Floods affect over 1 million in Balkans, destruction "terrifying"

May 20 2014: More than 40 killed in floods in Serbia, Bosnia, Croatia.

Figure 1-1: Headlines from Reuters.com

The United States Defence Advanced Research Projects Agency (DARPA) is responsible for the development of new technologies for use by the military and is currently funding extensive research work in this area (Do, Lu, and Roth 2012). There has been much recent progress in being able to temporally relate events in newsfeeds. Notably the adoption of the ISO TimeML standard in 2005 (Pustejovsky et al. 2005) which is based on the Allen Temporal Intervals (Allen 1983). TimeML provides the standard for temporally labelling the constituent parts of a newsfeed and its adoption has facilitated the compilation of corpora containing TimeML annotated newsfeed documents for the purpose of machine learning. The TempEval shared task was established in order to advance research in machine learning on temporally labelled newsfeed documents. There have been three resultant TempEval Challenges of increasing difficulty; TempEval-1 (Verhagen et al. 2007), TempEval-2 (Verhagen et al. 2010) and TempEval-3 (UzZaman et al. 2012). The next challenge will be held in Dublin in 2015. Task C of the most recent challenge focused on the detection and classification of the relationship types in a newsfeed. There were thirteen participant classifiers. Task C was different from previous challenges in that prediction using the full set of relationship types was included for the first time; a new 'Platinum' test base corpus containing hand-annotated newsfeeds was compiled for the challenge;

[1] United Nations Internationally agreed Glossary of basic terms related to Disaster Management

and a new temporal awareness measurement tool was developed in order to score the participant classifiers on their predictions.

The participants and organisers of the recent Challenge permitted that we use the resultant output from their corpus trained classifiers together with the measurement tool used for the Challenge.

A Mathematica analysis of the graphs produced by the hand-annotated and machine learned annotated newsfeeds revealed an interesting diversity in predicted relation across different classifiers. Figure 1-2 and Figure 1-3 contrast the hand-annotated bbc_20130322_1600 newsfeed with those predicted by three different classifiers each using a different machine learning technique. The aim of our research was to apply a statistical Integer Linear Programming (ILP) ensemble method to reconcile the predictions of the constituent classifiers subject to a global consistency check on the predicted relations. Our aim was to improve the temporal awareness score with a focus on the quality of the resultant prediction. An ensemble method has not been applied to this task previously. We aimed to union statistically weighted predicted arcs of the classifiers in order to more closely resemble the Platinum test hand-annotated newsfeed. The human annotators tend to produce graphs, which are almost regular, i.e. there is little variance in the degrees of the vertices. In contrast, the machine-learning methods often produce references to a single time-point in the past, which, although not wrong, are not the most accurate, which leads to very uneven distributions of degrees in the graph.

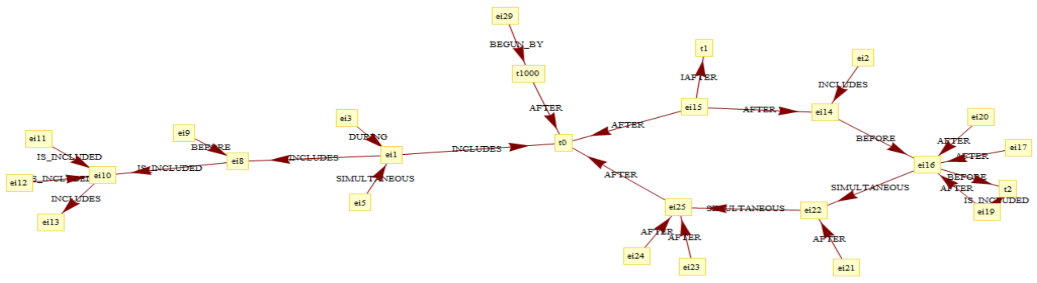

Figure 1-2: Platinum hand-annotated bbc_20130322_1600

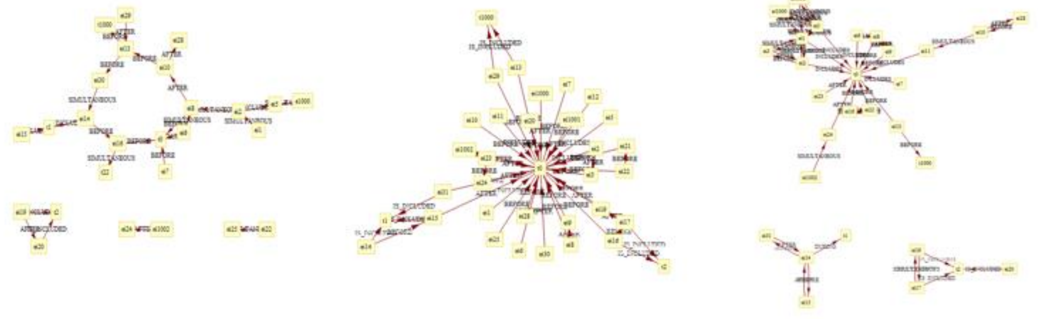

Figure 1-3: Classifier diversity bbc_20130322_1600

In order to inference and answer questions relating to the events, we need to be sure that the temporal relationship labelling is consistent and correct. A weakness in the machine learning approach is that while it can detect patterns enabling it to predict relations, it cannot check the consistency of these predictions. We required to combine the most likely relations on the arcs subjected to a global consistency check across the combined predicted relations. ILP is well suited to the task of resolving complex combinatorial problems and there is on-going research into improving existing algorithms in order to tackle increasingly complex combinatorial problems (Williams 2013)

Much of the research into temporally labelling documents is built on the Allen Temporal Intervals (Allen 1983). Time is represented by what can be infinitesimally small intervals with a set of thirteen mutually exclusive relations between intervals. There are six primitive relations; their converses and the equals relation. TimeML is an implementation of the Allen Intervals with a fourteenth relation to include the converse of equals.(Figure 1-4)

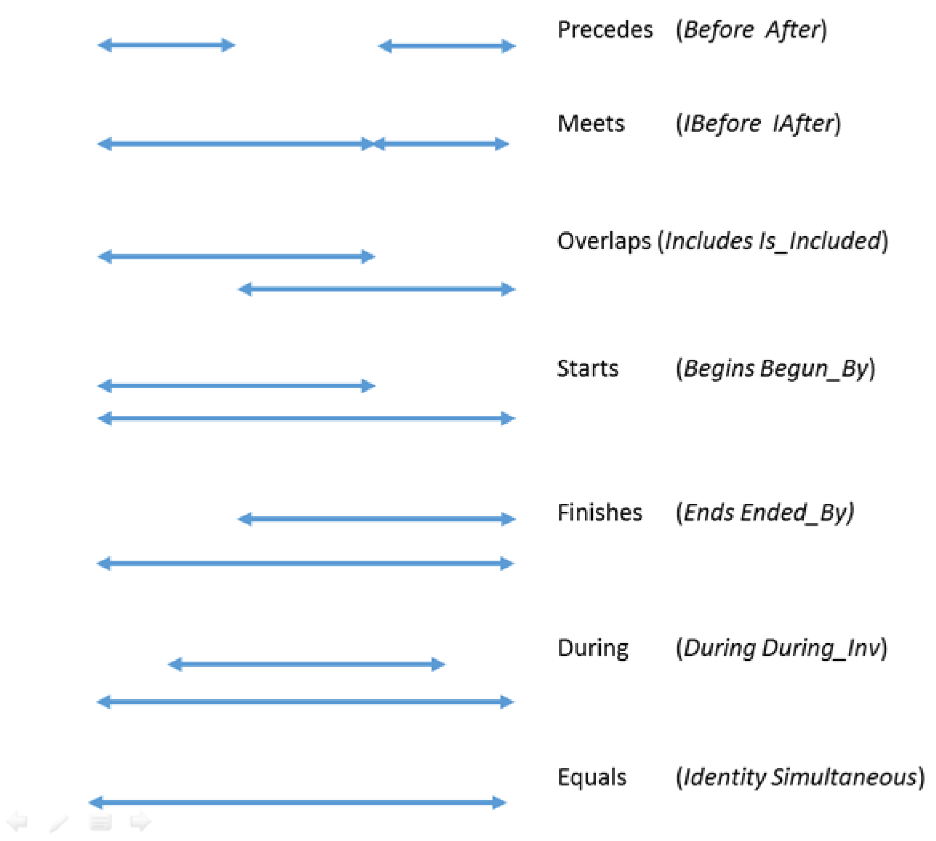
Precedes (*Before After*)
Meets (*IBefore IAfter*)
Overlaps (*Includes Is_Included*)
Starts (*Begins Begun_By*)
Finishes (*Ends Ended_By)*
During (*During During_Inv*)
Equals (*Identity Simultaneous*)

Figure 1-4: Allen's rules on transitive closure

The Allen Temporal Intervals introduced a Relation Algebra into temporal reasoning. The Algebra is based on the work of Alfred Tarski on relational calculus. To quote (Tarski 1941), on the application of relational calculus,

> "*Aside from the fact that the concepts occurring in this calculus possess an objective importance and are in these times almost indispensable in any scientific discussion, the calculus of relations has an intrinsic charm and beauty which makes it a source of intellectual delight to all who become acquainted with it*".

The theorem and axioms provide a necessary and sufficient condition for the temporal consistency of the transitive relations in a newsfeed document. An example of a transitive relation is depicted in Figure 1-5.

The **composition** or relative multiplication R o S of two relations is defined as:-
$R \circ S = \{< x, y >: (\exists z)[xRz \wedge zRy]\}$ implies the existence of some z with $xRz \wedge zSy$

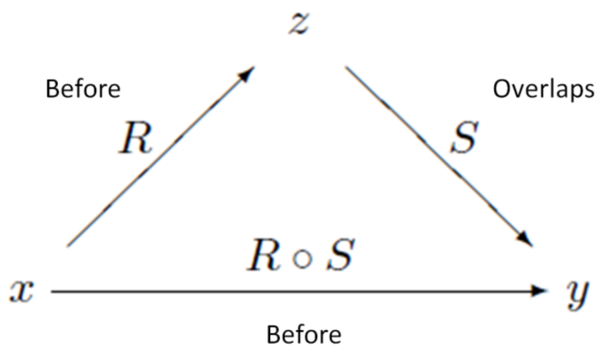

Figure 1-5: Transitive composition

We built and tested an ILP ensemble model using open-source software including Python (high level programming language with extensive libraries for handling complex data structures including the TimeML XML annotation format); Pyomo (a Python based mathematical modelling tool) and Cbc (a C++ branch and cut ILP optimizer). An amendment to a single line of code was required for final testing using IBM's commercial CPLEX optimizer.

The project required that we apply both the breadth and depth of knowledge acquired over an intensive year's taught master's programme. It required a multi-disciplinary approach guided by our IBM sponsor and UCD academic supervisor. There were times when a high level understanding was sufficient and others where in depth research was required in order to inform an approach to questions such as what makes a good ensemble, how do we measure an ensemble, can we apply the theory of a Bayes Optimal Ensemble Classifier to the task?

We are particularly grateful to the organisers and participants of the TempEval-3 Tack C Challenge for the use of their data on which this research is based. We have been privileged to have the opportunity to work in such an exciting area at this time.

A Review of the relevant literature is presented in Chapter 2; our Methodology is presented in Chapter 3; Measurements and Results in Chapter 4; Analysis and Discussion in Chapter 5; and our Conclusion in Chapter 6.

Chapter 2 - Literature Review

Our methodology was informed by a literature review of the current status of research into the temporal processing of newsfeeds. An overview of this literature is included in section 2.1. Our specific research aim within the context of this research further required reference to the literature across a number of scientific disciplines including logic as applied to global rules in section 2.2; machine learning ensemble technique in section 2.3; statistical methods for classifier prediction reconciliation in section 2.4; and the application of Integer Linear Programing in section 2.5.

2.1 Status of Research into Temporal Ordering of Events in newsfeeds

TimeEval Annotation Standard

Current research into temporal reasoning is based on the definition of the TimeML annotation standard as an ISO standard for temporally labelling documents in 2005.

TimeML (Pustejovsky et al. 2005) is an annotation scheme based on the Allen Intervals (Allen 1983). TimeML annotation is used for markup of events, times, and their temporal relations in news articles. This is accomplished using four primary tag types: TIMEX3 for temporal expressions, EVENT for temporal events, SIGNAL for temporal signals, and TLINK for representing relationships. The TimeBank Corpus (Pustejovsky et al. 2003) and Acquaint corpora contain over two hundred news articles that have been manually annotated according to the TimeML specification. Manual annotation of training data is very costly, not least because inter-annotator agreement for TLINKs is only 55% (average of Precision and Recall). Automated techniques to assist with annotation have been proposed (Cassidy et al.). The task of temporal labelling is to identify events and time expressions, link pairs of event-event (ordering) or event-time (anchoring) and label the relationship. We refer to (Saurı et al. 2009) for a full description of the TimeML annotation scheme.

Machine Learning and TempEval Challenges

The availability of an annotated corpus encouraged the application of machine learning techniques for the task of temporal labelling. (Mani et al. 2006) and (Bethard, Martin, and Klingenstein 2007) The TempEval shared task was established in order to advance research on temporal information processing. The TempEval

Challenge (Verhagen et al. 2009) is a framework for evaluating systems that automatically annotate texts with temporal relations using the TimeML annotation language. There have been three challenges to date; TempEval-1 (Verhagen et al. 2007), TempEval-2 (Verhagen et al. 2010) and TempEval-3 (UzZaman et al. 2012).The TempEval-3 Challenge included three tasks:

A. Time expression extraction and normalisation
B. Event extraction and classification
C. Detecting and annotating temporal relations

The focus of our project was Task C. There were thirteen participant classifiers for the TempEval-3 Task C Challenge using a variety of state of the art classifiers. (Chambers 2013) (Bethard 2013) (Laokulrat et al. 2013). It was also the first time that detection and classification of the full fourteen relations was included. A new temporal awareness tool (UzZaman et al. 2012) was introduced to the Challenge to measure *Precision*, *Recall* and overall *F1* score. These metrics are defined below and expanded on in Chapter 3. They measure how many entities (events and temporal expressions) are correctly identified.

$$Precision = \frac{Sys_{entity} \cap Ref_{entity}}{|Sys_{entity}|}$$

$$Recall = \frac{Sys_{entity} \cap Ref_{entity}}{|Ref_{entity}|}$$

Sys_{entity} are the entities extracted by the system, Ref_{entity} are the reference entities

$$F1(Recall, Precision) = 2 * \frac{Recall * Precision}{(Recall + Precision)}$$

The participants and organisers of the Challenge made both results from the Challenge as well as the temporal awareness measurement tool available for the purposes of our research into the use of an ensemble technique. All the TempEval-3 participant classifiers were trained on the TimeBank and Acquaint corpora. The machine learning strategies and techniques together with recorded temporal awareness scores are recorded in Table 2-1.

Participant	*Identif. Strategy*	*Identif. Classif.*	*Classif. Strategy*	*Identif. Classif.*	*F1*	*Precision*	*Recall*
UTTime1	Rules	None	Data	Logit	24.28	14.90	65.56
UTTime2	Rules	None	Data	Logit	24.15	14.87	64.24
UTTime3	Rules	None	Data	Logit	24.22	15.07	61.70
UTTime4	Hybrid	Logit	Data	Logit	28.82	37.52	23.40
UTTime5	Hybrid	Logit	Data	Logit	34.99	36.05	34.00
ClearTK1	Data	Logit	Data	SVM, Logit	35.17	37.64	33.00
ClearTK2	Data	SVM, Logit	Data	SVM, Logit	36.24	37.32	35.21
ClearTK3	Data	SVM, Logit	Data	SVM,	34.21	33.36	35.10
ClearTK4	Data	MaxEnt	Data	Logit	35.94	35.26	36.64
NavyTime1	Hybrid	MaxEnt	Data	MaxEnt	30.79	35.19	27.37
NavyTime2	Hybrid	MaxEnt	Data	MaxEnt	25.68	30.92	21.96

Table 2-1: TempEval-3 - participant data used for ensemble

Leveraging Machine Learning by Enforcing Global Constraints

An active area of research is that of applying global constraints to local predictions. (Tatu and Srikanth 2008) used global constraints to expand training data and identify temporal inconsistencies. They applied a greedy search to select the most appropriate configuration of temporal relations among events and time expressions. (Bramsen et al. 2006), (Chambers and Jurafsky 2008), (Talukdar, Wijaya, and Mitchell 2012) applied global constraints on one local classifier using ILP and subsets of the Allen relations. (Yoshikawa et al. 2009) formulated a joint inference model with Markov Logic Networks limited to pairs of temporal entities in the same or adjacent sentences. (Do, Lu, and Roth 2012) constructed an event timeline with time intervals using ILP to couple local event-event and event-time classifiers subject to global coherence constraints.

Our research work is the first time that both an ILP ensemble technique is employed to reconcile the results of multiple classifiers and the full set of fourteen relations is applied for global coherence checking.

2.2 Global Rules and Reasoning with Time

Towards Automated Reasoning with Time

Time-stamping of records stored in the databases supporting information systems is standard practice in order to retrieve information such as "which employees worked for us last year who made in excess of $60,000". In addition, as data is normalised (stored in multiple related tables) in relational database systems, retrieval of information requires that the system perform indexed joins across tables. It is imperative that these joins return a consistent view of the data for the related time period. Processing data in a newsfeed requires that inference be performed on unstructured data that has been annotated according to a standard (TimeML) format. Automating the retrieval of a temporally consistent view of the information is a much more difficult task. It requires that a necessary and sufficient condition of consistency across the annotated relations be in place.

Allen Intervals

(Allen 1983) was a landmark in temporal reasoning. He describes "*a system for reasoning about temporal intervals that is both expressive and computationally effective. The representations capture the temporal hierarchy implicit in many domains by using a hierarchy of reference intervals which precisely contain the amount of deduction performed automatically by the system*".

All events are represented by intervals of time that may be infinitesimally small. A binary relationship between any two intervals is represented by one and only one interval relation. There are thirteen relations:- six primitive relations; the converses of these six relations and the equals relation. The ISO TimeML defined standard discussed in 2.1 includes, in addition, a converse for the equals relation for consistency. The Allen Interval relations are depicted in Figure 2-1 with the (bracketed) related TimeML implemented standard.

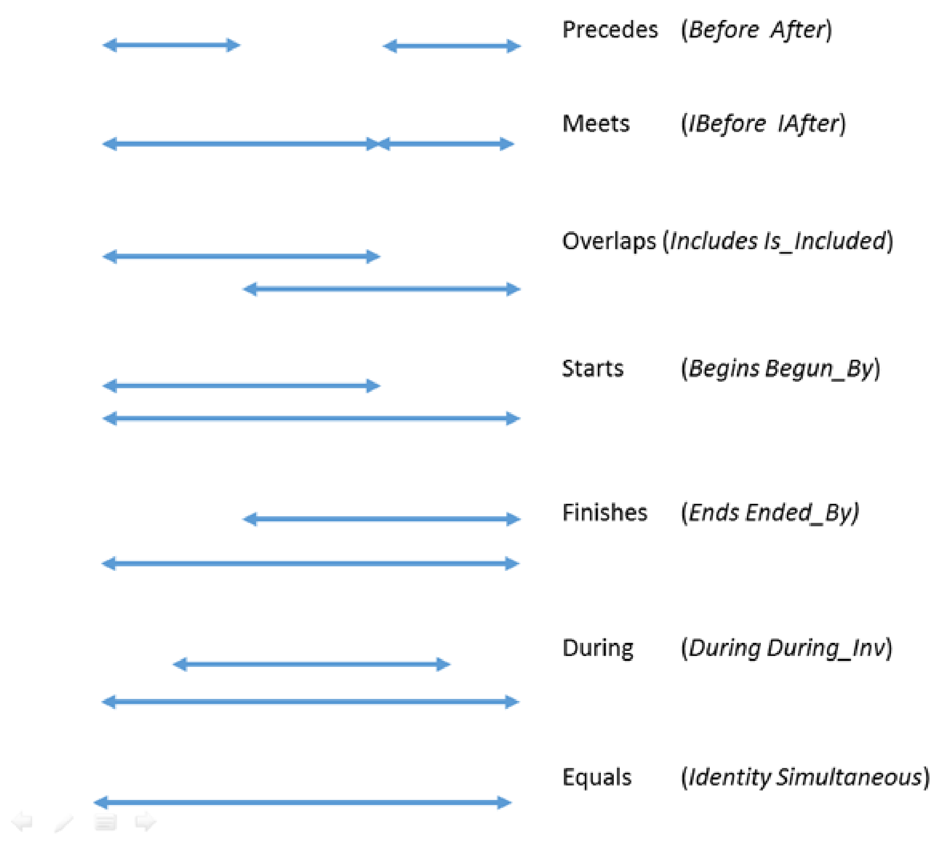

Figure 2-1: The Allen Interval Relations (TimeML)

Relation Algebra

Relations and their algebras were first introduced by Charles Sanders Peirce and Ernst Schröder in the nineteenth century and subsequently formulated into an algebraic framework by the renowned Alfred Tarski in the twentieth century (Tarski 1941).

A Relation Algebra is a binary relation algebra (Düntsch 2005). A binary relation R on a set U is a subset $U \times U$ that is a set of ordered pairs $<x, y>$ where $x, y \in U$. It is represented xRy.

$$\begin{tikzcd} x \arrow[r, "R"] & y \end{tikzcd}$$

Figure 2-2: A Binary Relation

Each relation R induces a mapping $U \to 2^U$ via the assignment $x \mapsto \{y : xRy\}$.

There are two operators (converse and composition) and a constant (identity). The composition or relative multiplication $R \circ S$ of two relations is defined as

$$R \circ S = \{<x, y>: (\exists z)[xRz \text{ and } zSy]\}$$

$$\begin{tikzcd}
 & z \arrow[dr, "S" ', "\text{Overlaps}"] & \\
x \arrow[ur, "R", "\text{Before}" '] \arrow[rr, "R \circ S", "\text{Before}" '] & & y
\end{tikzcd}$$

Figure 2-3: Transitive Composition

The Relation Algebra provides a necessary and sufficient condition for consistency across relations in an annotated newsfeed. (Ladkin 1990) states the related theorem:

"Let V be any set, and $A_0,\ldots A_{n-1}$ be a partition of $(V\text{x}V)$ into n disjoint relations. This partition is the set of atoms of a proper relation algebra on V if and only if the converse of every relation in the partition is also in the partition; the composition of any two relations in the partition is a union of sets in the partition; and the identity relation on V is a union of sets in the partition."

The (Allen 1983) composite table is presented Table 2-3. It has been mapped to the TimeML annotation standard for ease of reference and therefore also includes the converse of identity relation.

p	BEFORE	p̆	AFTER
o	INCLUDES	ŏ	IS-INCLUDED
d	DURING	d̆	DURING_INV
I	IDENTITY	Ĭ	SIMULTANEOUS
m	IBEFORE	m̆	IAFTER
s	BEGINS	s̆	BEGUN_BY
f	ENDS	f̆	ENDED_BY

Table 2-2: Legend for TimeML mapping of Allen's rules table

	I	**Ĭ**	**p**	**p̆**	**d**	**d̆**	**o**	**ŏ**	**m**	**m̆**	**s**	**s̆**	**f**	**f̆**
I	I	I Ĭ	p	p̆	d	d̆	o	ŏ	m	m̆	s	s̆	f	f̆
Ĭ	Ĭ I	Ĭ	p	p̆	d	d̆	o	ŏ	m	m̆	s	s̆	f	f̆
p	p	p	p		p d o m s	p	p	p d o m s	p	p d o m s	p	p	p d o m s	p
p̆	p̆	p̆		p̆	p̆ d ŏ m̆ f	p̆	p̆ d ŏ m̆f	p̆	p̆ d ŏ m̆f	p̆	p̆ d ŏ m̆f	p̆	p̆	p̆
d	d	d	p	p̆	d		p d o m s	p̆ d ŏ m̆f	p	p̆	d	p̆ d ŏ m̆f	d	p d o m s
d̆	d̆	d̆	p d̆ o m f̆	p̆ d̆ ŏ m̆ s̆	d d̆ s s̆ f f̆ o ŏ I Ĭ	d̆	d̆ o f̆	d̆ ŏ s̆	d̆ o f̆	d̆ ŏ s̆	d̆ o f̆	d̆	d̆ ŏ s̆	d̆
o	o	o	p	p̆ d̆ ŏ m̆ s̆	d o s	p d̆ o m f̆	p o m	I Ĭ d d̆ o ŏ s s̆ f f̆	p	d̆ ŏ s̆	o	d̆ o f̆	d o s	p o m
ŏ	ŏ	ŏ	p d̆ o m f̆	p̆	d ŏ f	p̆ d̆ ŏ m̆ s̆	I Ĭ d d̆ o ŏ s s̆ f f̆	p̆ ŏ m̆	d̆ o f̆	p̆	d ŏ f	p̆ ŏ m̆	ŏ	d̆ ŏ s̆
m	m	m	p	p̆ d̆ ŏ m̆s̆	dos	p	p	d o s	p	I Ĭ f f̆	m	m	dos	p
m̆	m̆	m̆	p d̆ o m f̆	p̆	d ŏ f	p̆	d ŏ f	p̆	i Ĭ s s̆	p̆	d ŏ f	p̆	m̆	m̆
s	s	s	p	p̆	d	p d̆ o m f̆	p o m	d ŏ f	p	m̆	s	I Ĭ s s̆	d	p o m
s̆	S̆	s̆	p d̆ o m f̆	p̆	d ŏ f	d̆	d̆ o f̆	ŏ	d̆ o f̆	m̆	I Ĭ s s̆	s̆	ŏ	d̆
f	f	f	p	p̆	d	p̆ d̆ ŏ m̆s̆	d o s	p̆ ŏ m̆	m	p̆	d	p̆ ŏ m̆	F	I Ĭ f f̆
f̆	f̆	f̆	p	p̆ d̆ ŏ m̆ s̆	d o s	d̆	o	d̆ ŏ s̆	m	d̆ ŏ s̆	o	d̆	I Ĭ f f̆	f̆

Table 2-3: TimeML mapping of Allen's rules on time interval relations

Temporal Path Consistency and Inference

Allen was the first to identify practical consistency checking techniques that used the thirteen interval relations (Ladkin 1990). A path consistency algorithm is an algorithm that iterates over all possible triangles in the graph formed by an annotated newsfeed document, performing composition consistency checks on each edge. Path consistency says that the label on any edge in the graph is contained in the composition along any path beginning at the tail of the edge and ending at the head. 3-consistency is path consistency for paths of length 2. It has been shown that 3-consistency is equivalent to the seemingly more general concept of path consistency.

The Allen interval end points may be mapped to a discrete time linear < or = representation for discrete time inference per Table 2-4.

TimeML	Allen	End Points	Discrete time <=	
BEFORE	precedes	{<[q,r],[q’,r’]> }	$q < r < q' < r'$	$q, r, q', r' \in R$
IBEFORE	meets	{<[q,r],[q’,r’]> }	$q < r = q' < r'$	$q, r, q', r' \in R$
INCLUDES	overlaps	{<[q,r],[q’,r’]> }	$q < q' < r < r'$	$q, r, q', r' \in R$
BEGINS	starts	{<[q,r],[q’,r’]> }	$q = q' < r < r'$	$q, r, q', r' \in R$
ENDS	finishes	{<[q,r],[q’,r’]> }	$q' < q < r = r'$	$q, r, q', r' \in R$
DURING	during	{<[q,r],[q’,r’]> }	$q < q' < r' < r$	$q, r, q', r' \in R$

Table 2-4: TimeML-Allen Interval End Points Discrete Time Linear Mapping

2.3 *Ensemble Methods in Machine Learning Ensemble*

Ensemble methods are learning algorithms that look to improve on overall predictive accuracy by taking a weighted average over the predictions of a number of diverse classifiers.

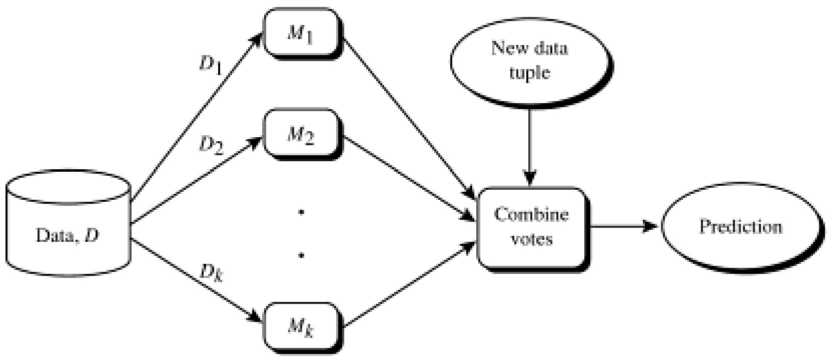

Figure 2-4: Ensemble methods

(Han and Kamber 2011) Ensemble methods generate a set of classification models, $M_1, M_2, ..., M_k$.Given a new data tuple to classify, each classifier "votes" for the class label of that tuple. The ensemble combines the votes to return a class prediction. Properties of ensemble methods include:-

- an ensemble tends to be more accurate than its base classifiers
- an ensemble yields significantly better results when there is significant diversity with little correlation among the component classifiers
- component classifiers should perform better than random guessing for best results
- as each base classifier can be allocated to a different CPU, ensemble methods are parallelizable greatly enhancing the scalability of the method applied to complex problems

(Dietterich 2000) outlines reasons for the effectiveness of ensemble methods as being:-

- Statistical

  A learning algorithm can be viewed as searching an hypothesis space H in order to identify the best hypothesis in the space. When the training data sample is too small, the learning algorithm can find many different hypotheses in H that all give the same accuracy on the training data. An ensemble can "average" the classifier votes and reduce the risk of choosing the wrong classifier.

- Computational

  Many learning algorithms perform some form of local search that may get stuck in local optima. An ensemble may provide a better global approximation

- Representational

  In most applications of machine learning, the true prediction cannot be represented by any of the hypotheses in H. By forming a weighted union drawn from H, it may be possible to extend the space of representable predictions.

Currently an active research area, types of ensemble methods that have been applied for different applications include Bayesian Voting; Manipulating Training Examples (Bagging and Boosting); Manipulating Input Features; Manipulating Output Targets; and Injecting Randomness (Random Forests). An outline of the methods researched that formed our approach is included below. Our project is the first example of the application of ensemble methods to the problem of detecting and classifying temporal relations in newsfeeds. It is further differentiated in that ILP is used as a method of ensemble.

Bayesian Voting: Enumerating the hypotheses (Dietterich 2000)

In a Bayesian probabilistic setting, each hypothesis h defines a conditional probability distribution:

$$h(x) = P(f(x) = y \mid x, h)$$

Given a new data point x and a training sample S, the problem of predicting the value of $f(x)$ can be viewed as the problem of computing

$$P(f(x) = y \mid S, x)$$

We can rewrite this as weighted sum over all hypotheses in H:

$$P(f(x)=y \mid S,x)=\sum_{h\in H} h(x)P(h \mid S)$$

The above can be viewed as an ensemble method in which the ensemble consists of all of the hypotheses in H, each weighted by its posterior probability $P(h \mid S)$. Applying Bayes rule, the posterior probability is proportional to the likelihood of the training data times the prior probability of h

$$P(h \mid S) \propto P(S \mid h)P(h)$$

For some learning problems, it is possible to completely enumerate each $h \in H$, compute $P(S \mid h)$ and $P(h)$ and evaluate this Bayesian "committee." In addition, if the true function f is drawn from H according to $P(h)$, then the Bayesian voting scheme is optimal.

Bayesian voting primarily addresses the statistical component of ensembles. When the training sample is small, many hypotheses h will have significantly large posterior probabilities, and the voting process can average these to "marginalize away" the remaining uncertainty about f.When the training sample is large, typically only one hypothesis has substantial posterior probability, and the ensemble effectively shrinks to contain only a single hypothesis.

In complex problems where H cannot be enumerated, it is sometimes possible to approximate Bayesian voting by drawing a random sample of hypotheses distributed according to $P(h \mid S)$.

The key and most idealized aspect of Bayesian analysis is the prior belief $P(h)$. If this prior completely captures all of the knowledge that we have about f before we obtain S, then by definition we cannot do better. However, in practice, it is often difficult to construct a space H and assign a prior $P(h)$ that captures prior knowledge adequately. In such cases, the Bayesian "committee" is not optimal, and other ensemble methods may produce better results.

Bagging: Bootstrap Aggregating (Han and Kamber 2011)

Intuitively a majority vote made by a large group of classifiers may be more reliable than the majority vote made by a small group. The increased accuracy occurs because the composite model reduces the variance of the individual classifiers.

Given a set, D, of tuples, for iteration $i(i = 1, 2, ..., k)$, a training set D_i of d tuples is sampled with replacement from the original set of tuples, D. Each training set is a bootstrap sample. As sampling with replacement is used, some of the original tuples of D may not be included in D_i , whereas others may be included more than once. A classifier model M_i, is learned for each training set D_i. To classify an unknown tuple, X , each training classifier M_i , returns the class prediction which counts as one vote. The bagged classifier, M^* counts the votes and assigns the class with the most votes to X .

Boosting (example AdaBoost) (Han and Kamber 2011)

Intuitively instead of relying on one classifier, you choose to consult several and assign weights to the worth of each classifier's prediction based on the measured accuracy score from previous predictions made by the classifier. The final prediction is then a combination of the weighted predictions.

In boosting weights are also assigned to each training tuple. A series of k classifiers is iteratively learned. After a classifier, M_i is learned, the weights are updated to allow the subsequent classifier, M_{i+1} to "pay more attention" to the training tuples that were mis-classified by M_i . The final boosted classifier, M^* combines the votes of each individual classifier, where the weight of each classifier's vote is a function of its accuracy.

Our ensemble method was inspired by techniques used as part of the methods above. We implemented a set of experimental formulae to combine sets of arcs across classifiers; to apply classifier accuracy dependant weights and to incorporate Bayesian prior knowledge of the distribution of the relationship types on the training datasets. In particular, the specification of a strong Bayesian prior on the distribution of relationship types proved difficult limiting the effectiveness of a statistical Bayesian voting approach.

"A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse" (Dietterich 2000). Measurements relating to accuracy and diversity were performed on the data up front and are included in 3.2.

2.4 Classifier Reconciliation and a Probabilistic Approach

Pattern recognition or discrimination involves making predictions based on observations. In our case, the predictions are the temporal relationships existing in a newsfeed and the observations are the individual predictions of a set of classifiers using state of the art methods. Obvious questions include: what theory can we apply to build an 'optimal ensemble classifier' and is there a way of measuring the probability of error of this 'optimal ensemble classifier'? The Theory of a Bayes Optimal Classifier addresses the former question while that of a Bayes Error addresses the latter.

In (Devroye 1996) and statistical machine learning in general an *observation* is a d-dimensional vector x. There is a finite set of M classes. The unknown nature of an observation is described by a mapping from observations to *classes.* In pattern recognition, one creates a function $g(x): R^d \rightarrow \{1,\ldots,M\}$ which represents one's guess of class y given x. The mapping g is called a *classifier.* The classifier errs on x if $g(x) \neq y$.

For some x, such as the temporal relations in a newsfeed, it is in fact impossible to specify g for all possible cases of x. We have to accept imperfect g which introduces probability in the Bayesian sense of 'degree of belief' in the predictions.

(Devroye 1996) introduce a probabilistic setting. Let (X,Y) be an $R^d \mathrm{X}\{1,\ldots,M\}$-valued random pair. The distribution of (X,Y) describes the frequency of encountering particular pairs in practice. An *error* occurs if $g(x) \neq y$, and the *probability of error* for a classifier g is $L(g) = P\{g(\mathrm{X}) \neq \mathrm{Y}\}$. There is a best possible classifier g^* which is defined by

$$g^* = \underset{g:R^d \rightarrow \{1,\ldots M\}}{\arg\min} P\{g(X) \neq Y\} .$$

The problem of finding g^* is Bayes problem and the classifier g^* is the Bayes Optimal Classifier. The minimal probability of error is called the Bayes Error and is denoted by $L^* = L(g^*)$. In most cases the distribution of (X,Y) is unknown, so that g^* is also unknown. We do, however, have access to data pairs $(X_{i,} Y_i), 1 \leq i \leq n$ which can be used to construct a classifier g.

A classifier constructed on the basis of $X_1, Y_1, ..., X_n, Y_n$ is denoted by g_n. A sequence $\{g_n, n > 1\}$ is called a (discrimination) *rule.* Thus classifiers are functions and rules are sequences of functions. A *rule* is good if it is *consistent* that is

$$\lim_{n \to \infty} EL_n \to L^*$$

Our aim within the context of the project was to build a tool so that we could experiment in finding a consistent rule.

(Devroye 1996) proposes that a consistent rule guarantees that by increasing the amount of data (taking into account the predictions of more classifiers) the probability that the error is within a very small distance of the optimal achievable gets arbitrarily close to one. Intuitively, the rule can eventually learn the optimal decision (correct temporal labelling) from a large amount of training data with high probability. A Bayesian approach depends on a good prior estimate being available of the distribution of relationship types.

The *rule* composition for experimentation is made up of:-

- a maximum-margin weighting on classifier accuracy scores based on either proportion of classifiers or maximum-likelihood, using knowledge of prior distribution of the relationship types on the training dataset.
- a loss function that seeks to find the function that minimises the cost of error
- an integer programming (IP) formulation that enforces global temporal relationship consistency checks within and across classifiers
- knowledge of classifier diversity, in our case compiled from Mathematica and an analysis of the graphs produced by individual classifiers. (See Figure 3-1 and Table 3-2: *Summary of classifier diversity*.

In deciding on the formulae for experimentation, we were informed by Bayesian theory on maximum likelihood and inclusion of prior knowledge as well as decision theory on the application of loss functions.

Maximum Likelihood

The use of the Maximum Likelihood Principle in pattern matching (discrimination) problems is only applicable if we have some prior knowledge of the problem (Devroye 1996). In our case prior knowledge included the accuracy scores of the classifiers and the distribution of the different relationship types on the training datasets. Maximum likelihood aims make the known likelihood distribution a maximum when assigning probabilities based on the observed data.

Prior distributions and a formula for a Bayes Optimal Classifier

The formula for a Bayes Optimal Classifier takes account of *prior* probabilities in the calculation of *posterior* probabilities.

$$\arg\max_{c_i} \sum_{h_i} P(c_i \mid h_i) P(T \mid h_i) P(h_i)$$

We could obtain estimates for $P(c_i \mid h_i)$ and $P(h_i)$. However, estimating $\mathrm{P(T|h}_i)$ proved difficult in implementing above.

We allow for consideration of prior distributions in weighting formulae and iterate over the all the hypotheses H (probability of each relation, BEFORE, AFTER, etc.) given by feasible solutions to the ILP, i.e. not violating any constraints. Our hypotheses are multi-variate. We combine uni-variate c_i in that we consider the classification suggested by classifier j for an arc z and classification suggested by classifier k for arc z.

A different perspective, counting the cost of error and applying a loss function

(Larson 1982) Decision Theory postulates the existence of a loss function $l(\theta, g)$ which for every possible true value of θ and estimate g, expresses the loss suffered. We would, of course, like to find the g that minimizes loss suffered. The Bayesian estimate of an unknown parameter θ is the value of g that minimises $E[l(\theta, g)|x]$, the posterior expected value for the loss function. A loss function is used to weigh the cost of error. For example if you purchase a stock and the value goes down, you lose the full amount; if you did not buy the stock and the value goes down you have not lost anything but if you did not purchase the stock and the value goes up you incur a small penalty. Typical loss functions include 0-1 (discrete); hinge loss; log loss; square loss.

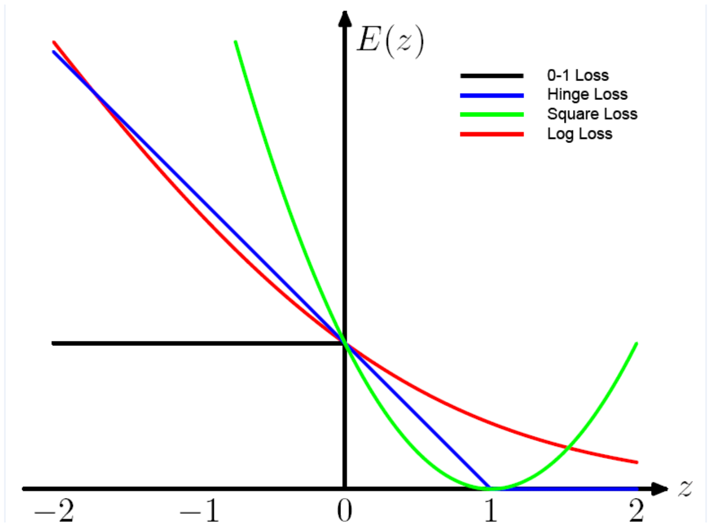

Figure 2-5: Loss functions (Bishop 2006)

2.5 Integer Linear Programming applied to Combinatorial Complexity

The objective of Natural Language Processing (NLP) is to enable computers to process and understand human languages. This is typically achieved through machine-learning techniques and probabilistic inference. A weakness of the machine-learning technique is that it does not recognise general (non-sequential) constraints. Much research has been completed using Integer Linear Programming (ILP) formulations to impose additional global consistency constraints. These constraints are typically implemented using indicator variables that can only take the binary values zero or one.

Formulations used for previous NLP research have included bi-partite assignment (Punyakanok et al. 2004), (Roth and Yih 2004), (Chambers and Jurafsky 2008); shortest path (Roth and Yih 2004) and graph colouring (Burke et al. 2010). The different approaches are outlined followed by a review of factors to consider in selecting a formulation for an ILP model (Williams 2013). Our approach, as informed by the literature review, is then discussed.

Formulations applied in previous research

(Roth and Yih 2004) examine the problem of entity and relationship recognition in text. Text is processed through a multi-class classifier, using a machine-learning algorithm. The output is the estimated probability distribution of the entity and relation class labels. An inference algorithm in the form of ILP formulations is applied to improve the accuracy of the class labels. The objective function is to minimise the cost of deviating from the assigned classes and the cost of breaking the constraints.

Minimise: $$C(f)=\sum_{u\in V} c_u(f_u)+\sum_{R_{ij}\in R}\left[d^1\left(f_{R_{ij}},f_{E_i}\right)+d^2\left(f_{R_{ij}},f_{E_j}\right)\right]$$

$c_u(f_u) = -\log$(probability of variable u having the assigned label)

$$d^1\left(f_{R_{ij}},f_{E_i}\right)=0 \text{ if } \left(f_{R_{ij}},f_{E_i}\right)\in C^1, \text{ otherwise Infinity}$$

C^1 is the set of permitted relationships for the first entity E_i d^1 ensures the relationship is consistent in the first argument. Similarly, d^2 forces consistency in the second argument.

They report that the formulation provided an efficient way of finding an optimum solution.

(Punyakanok et al. 2004) adopt a similar approach to the problem of semantic role labelling (SRL). The inference procedure takes confidence scores of each argument given by the classifier as input, and outputs the *best* global assignment that satisfies the constraints. The objective function is to maximise the overall score of the arguments.

$$argmax_{z\in\{0,1\}^d}\sum_{i=1}^{M}\sum_{c=1}^{|P|}p_{ic}z_{ic}$$

subject to

$$\sum_{c=1}^{|P|} z_{ic} = 1 \qquad \forall z_{ic}\in Z \qquad (1)$$

M is the set of arguments;

P is the set of labels;

p_{ic} is the probability of label c being assigned to argument i;

z_{ic} is 1 if label c is assigned to argument i, 0 otherwise

Constraint (1) ensures that each argument can only be assigned one class label.

Other global constraints can be included. For example, “within a sentence, a predicate may only take one subject (A0 label)”, is formulated as:

$$\sum_{i=1}^{M} z_{iA0} \leq 1$$

They note that although solving an ILP problem is hard in general, they are able solve the inference problem for 4305 sentences on a Pentium-III 800Mhz machine in under ten minutes.

(Roth and Yih 2005) extend the work of (Punyakanok et al. 2004) by applying a shortest path formulation to the SRL problem. Conditional Random Fields are linear-chain Markov Random Fields and the task of SRL is to identify segments of consecutive words in a sentence and classify them into one of several classes. This can be represented in graph form as a trellis and solved using the Viterbi algorithm (an efficient dynamic programming algorithm). The drawback is that constraints can only be applied to adjacent tokens in the sentence (the Markov principle states that the probability of the next state depends only on the current state). (Roth and Yih 2002) This problem is overcome by using ILP, which allows the inclusion of more *expressive* constraints. For example, "if label a appears, then label b must also appear" can be formulated as:

$$\sum_{0 \leq y \leq m-1} x_{i,ya} \leq \sum_{\substack{0 \leq y \leq m-1 \\ 0 \leq i \leq n-1}} x_{i,yb} \qquad \forall i, 0 \leq i \leq n-1$$

They note that although the addition of global constraints means that the linear programming relaxation does not guarantee the integer solution guaranteed by the standard shortest path formulation, the problem is still solvable by commercial ILP solvers fairly efficiently despite the number of variables and constraints. They attribute this to the sparseness of the constraint matrix.

(Chambers and Jurafsky 2008) apply ILP to the task of reconciling globally inconsistent pairwise temporal labels in a document. For example, given A Before B; B before C; and A After C, confidence scores are assigned and an optimal label assignment determined based on global consistency rules. The ILP formulation is stated

$$\max \sum_{i} \sum_{j} p_{ij} x_{ij}$$

Subject to

$$\forall i \forall j \; x_{ij} \in \{0,1\}$$
$$\forall i \; x_{i1} + x_{i2} + \ldots + x_{im} = 1$$

A transitivity constraint is added for all connected pairs i, j, k for each relation that infers c given a and b

$$x_{ia} + x_{jb} - x_{kc} \leq 1$$

They implement both greedy best-first and ILP and find that ILP performs better. They identify methods to identify unknown events as important in improving application of global constraints.

(Burke et al. 2010) examine the possibility of using a graph colouring algorithm to resolve temporal relationships. Seven integer programming (IP) formulations of vertex colouring are examined and a new formulation using “supernodes”[2] is proposed, where Q is a reversible clique partition and $q \in Q$ is a supernode.

$$x_{q,c} = \begin{cases} 1 \text{ if colour } c \text{ is included in the set assigned to } q \in Q \\ 0 \text{ otherwise} \end{cases}$$

In addition, (Burke et al. 2012) suggest that performance can be improved by the addition of “soft” constraints. The goal is to minimise the number of violations of the soft constraints. The method uses dependent variables whose values are derived from the values of the decision variables in the process of solving. In dense temporal relationship graphs, the high number of connected triplets may result in contradictory resolutions to transitivity constraints and the IP formulation would benefit from the relaxation of these constraints to soft ones.

Factors to consider in selecting a formulation

(Williams 2013) Despite ongoing theoretical and computational progress in ILP, when considering formulations to use when building ILP models particular attention should be given to the greater computational difficulty in solving ILP models over Linear Programming (LP) models.

Exploiting special structure

[2] A “supernode” is a complete subgraph, within which every pair of vertices have the same neighbourhood outside of the subgraph.

Exploiting structure inherent in the problem in the choice of formulation can sometimes allow the problem to be solved as a LP. Problems containing special structure include the transportation problem; cost network flow problem and assignment problem. If the structure is totally unimodular, the optimal solution of the corresponding LP problem always results in integer values for the integer variables and will allow the problem to be solved using LP methods.

Re-formulating the problem

In cases where the problem does not have an inherent special structure, it is often possible to reformulate the model or add additional constraints giving another model which is easier to solve.

As illustrated in Figure 2-6, the aim is to reformulate the ILP model such that the feasible region of the corresponding LP model ABCD becomes PQRSTUV., the convex hull or smallest convex set containing all the feasible integer points.

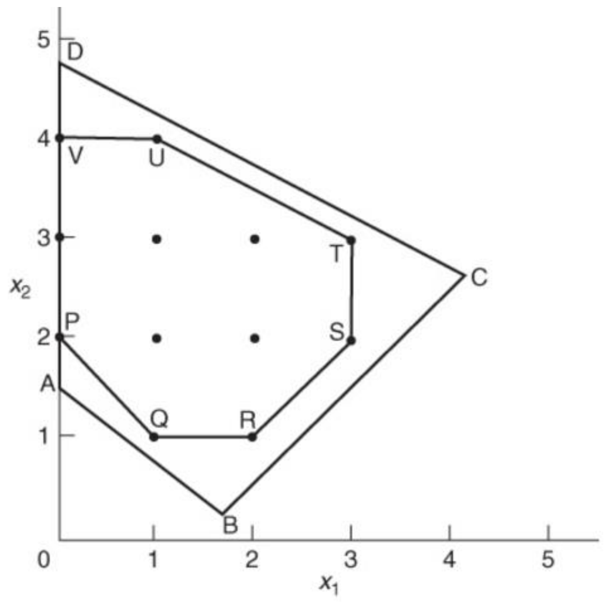

Figure 2-6: Convex Hull of Feasible Integer Solutions (Williams 2013)

Number of variables and number of constraints

The number of integer variables in an ILP model is often regarded as a good indicator of computational difficulty. For a model with *n* (0-1) variables this would indicate 2^n possible settings which increases exponentially with n. However, in practice, using the ILP branch and bound method (dividing an ILP problem into several LP sub-problems) many of these will not be examined. In fact, one may solve a 100 (0-1) variable ILP problem where only 0.00…01 (28 zeros after the decimal point) of the potential 2^{100} need be examined (Williams 2013). Therefore the number of 0-1 variables is often a poor indicator of the difficulty of an ILP model.

The number of constraints is also not a good indicator of difficulty. While the difficulty of an LP model is very dependent on number of constraints, an ILP is often made easier to solve by the addition of constraints.

Build a small test model

Build a small experimental model.

Our approach

We modelled the ensemble of the annotations of a single newsfeed by multiple classifiers as a directed multi-graph $G(V,E)$ with nodes V representing temporal events and temporal expressions and edges E representing the union of the TLINK relationships identified by the ensemble of classifiers between pairs of events and time expressions in V. The problem is that of determining an optimised Eulerian trail G' (V, E') on the graph G (V, E) that minimises the probability of mis-classification and most closely resembles the gold standard hand-annotation.

For implementation, we decided on a bi-partite assignment formulation that extended the formulation (Chambers and Jurafsky 2008) that implements transitivity on BEFORE and its converse AFTER to include the full set of transitivity constraints on fourteen relations. As we were implementing the complete set of fourteen relations, tractability and scalability were important considerations. While our formulation is not unimodular, (Roth and Yih 2004); (Roth and Yih 2005); and (Punyakanok et al. 2004) report efficient ILP implementations for similar formulations. The sparsity of the structure of the classifier annotation allowed us to minimise the number of variables to (no_arcs * 15) and the number of constraints to (no_connected_arcs * 15 * 15). The number of connected arcs was easily built up within the Python Pyomo modelling tool using the Python dictionary data structure with an index on arcs that allowed an O(1) search for existence of an arc. A Mathematica analysis of the structure of the graphs produced by the annotations showed an average density measure of 0.0629 and transitivity measure (no. closed triplets / no. connected triplets) of 0.2342.

(Hart et al. 2012) We implemented the experimental model using Coin-OR software. Coin-OR is a project that aims to create for mathematical software what the open source literature is for mathematical theory. To build our experimental model, we used Coopr Pyomo, a python package that combines the capabilities of a high-level programming language with the mathematical modelling capabilities required to formulate complex optimisation problems. We tested the model using Cbc (**C**oin-or **b**ranch and **c**ut), a mixed integer programming solver written in C++. For production implementations, commercial solvers such as CPLEX can be invoked from the Pyomo software using the same calling structure as that used for invoking Cbc.

Chapter 3 - Methodology

The aim of the work is to explore the effectiveness of an ensemble of classifiers in improving temporal relation detection and classification in text. We were able to utilise the data and tools that were used in the TempEval-3 challenge[3]. Our methodology addressed six key areas:

1. Data Management
2. Classifier diversity
3. Data processing pipeline
4. Weighting formulas
5. The ILP Model
6. The Measurement process is described in Chapter 4 - Measurements and Results.

3.1 Data Management

Data was kindly made available to us by the participants in the TempEval-3 challenge. This data consisted of a set of twenty annotated newsfeeds for each classifier and a "Platinum" set of the same newsfeeds, manually annotated by experts and which formed our benchmark against which the ensembles were measured. Each classifier's data was stored in a separate directory, along with a set of scores for the classifier. These scores were used to weight the votes of the individual classifiers. The scores were calculated using the same temporal evaluation tool that was used in the TempEval-3 challenge. The classifiers were assigned numbers (in the order in which we received the data) and these numbers are used throughout this document to describe the composition of the ensembles. See Table 3-1. For example, an ensemble described as 2-9-5-4 is composed of ClearTK-2, UTTime-4, Navytime-1, ClearTK-4.

[3] There have been several challenges based on the TimeBank corpus including the Temp-Eval-3 challenge (Verhagen et al. 2009) where participants have tested machine learning (SVM) and rule-based methods to advance research on temporal language processing.

Classifier number	Classifier name
1	ClearTK-1
2	ClearTK-2
3	ClearTK-3
4	ClearTK-4
5	Navytime-1
6	UTTime-1
7	UTTime-2
8	UTTime-3
9	UTTime-4
A	UTTime-5
B	Navytime-2

Table 3-1: Classifier identifiers

3.2 Classifier Diversity

Ensemble theory suggests that "*a necessary and sufficient condition for an ensemble of classifiers to be more accurate than any individual members is if the individual constituent classifiers are accurate and diverse*" (Dietterich 2000). Classifiers that are diverse make different errors. We carried out an analysis of the eleven classifiers at our disposal to ascertain whether they were diverse enough to cover the solution space. The first exercise was to do a pair-wise comparison of the classifiers. We measured the proportion of TLINKs that were classified differently by each classifier. Table 3-2 is a summary of these results. It is not an exhaustive pair-wise list, but it demonstrates that classifiers from different groups show a good level of diversity. The classifier groupings are 1-2-3-4 (ClearTK), 6-7-8 (UTTime-1,2,3), 9-A (UTTime-4,5), 5-B (Navytime-1,2). For example, ClearTK-1 and Navytime-2 are very diverse, but UTTime-1 and UTTime-2 are highly correlated.

Classifiers	Different classes assigned	Total classes assigned	Diversity measure
1-2	117	1043	11.22%
1-3	420	1210	34.71%
1-4	342	1181	28.96%
1-B	1262	1378	91.58%
2-5	1429	1569	91.08%
2-6	3569	4413	80.87%
2-9	1062	1320	80.45%
4-7	3487	4413	79.02%
4-9	1130	1401	80.66%
4-A	1163	1559	74.60%
5-9	941	1072	87.78%
5-B	348	736	47.28%
6-7	570	4413	12.92%
6-B	4222	4510	93.61%
7-9	3938	4413	89.24%
A-B	1105	1249	88.47%

Table 3-2: Summary of classifier diversity

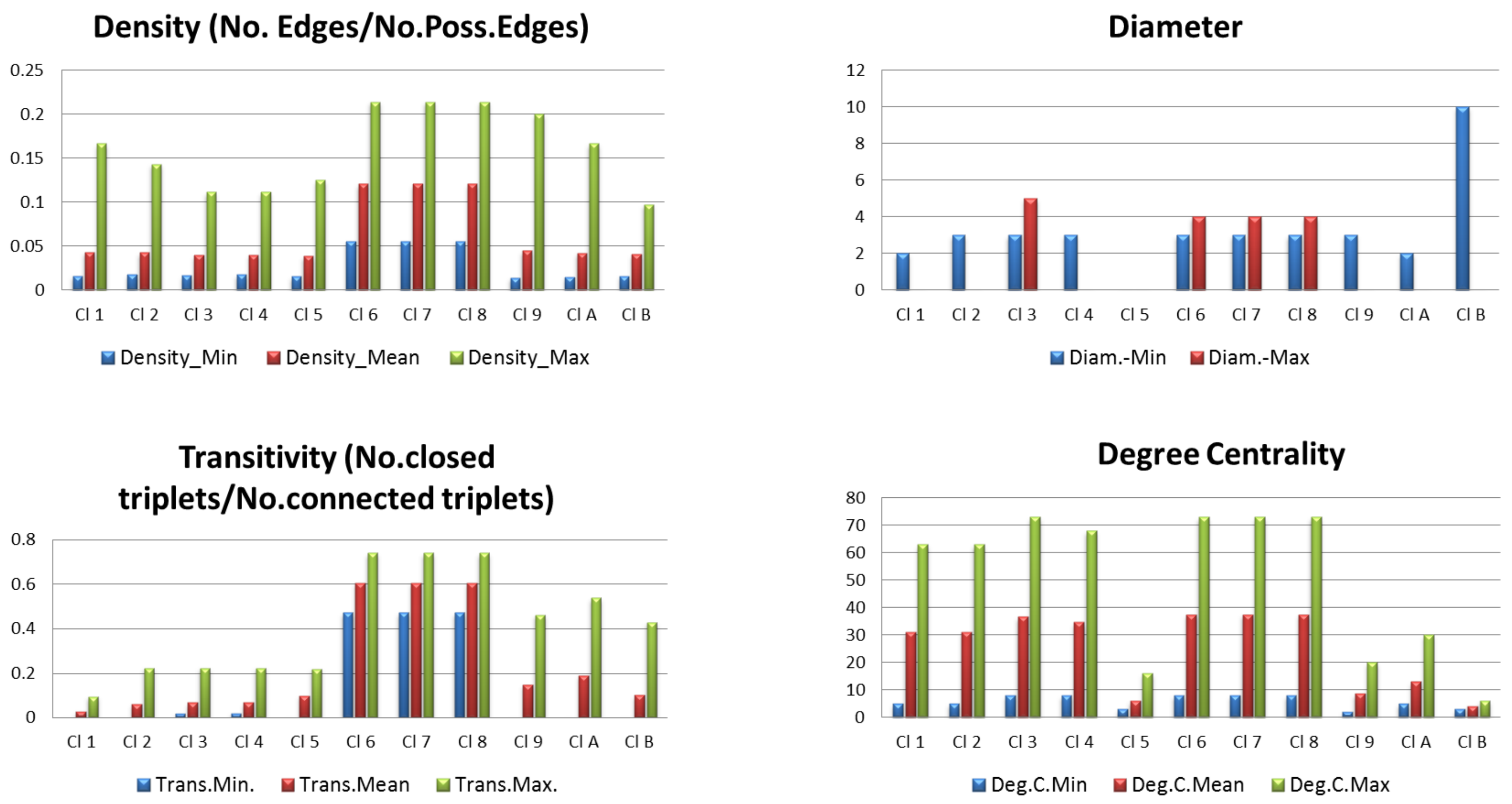

Figure 3-1: Graphs across 20 newsfeeds (min; max; mean)

We also performed a Mathematica[4] analysis of the structure of the newsfeed graphs.

The graphs in Figure 3-1 show the minimum, mean, and maximum measurements across the twelve newsfeeds for each of the eleven classifiers. These measurements can be interpreted with respect to the TimeML data as follows:

- Density: number of TLINKs divided by total possible number of TLINKs;
- Diameter: farthest distance between two vertices in a graph; 0 for an unconnected graph. This represents the farthest distance between two events connected by a chain of TLINKs.
- Transitivity: number of closed triplets over number connected triplets – global clustering coefficient. This represents the number of TLINKs that have been *explicitly* inferred by the classifier from pairs of connected TLINKs.
- Max degree centrality: maximum number of connections to any vertex representing event/time intervals.

This structural representation gives some insight into the diversity of the classifiers. Notice how the max degree centrality for the ClearTK classifiers (1-4) is very high compared with the density. This indicates these classifiers are strongly clustered around event/time-interval hubs. The UTTime classifiers (6-7-8), with high density and high transitivity, contain a high number of inferred TLINKs . The Navytime classifiers (5 and B), with high diameter, are less likely to have hubs and more likely to have chains of linked events/time-intervals .

We also carried out an analysis of the classifiers and the benchmark Platinum data to ascertain the highest F1, Precision and Recall scores[5] that are possible with the test data. This is explained in 4.1.

[4] Mathematica is a computational software program used in many scientific, engineering, mathematical and computing fields, based on symbolic mathematics.

[5] F1, Precision and Recall scores are temporal awareness metrics, as measured by the TempEval-3 measurement tool.

3.3 The Data Processing Pipeline

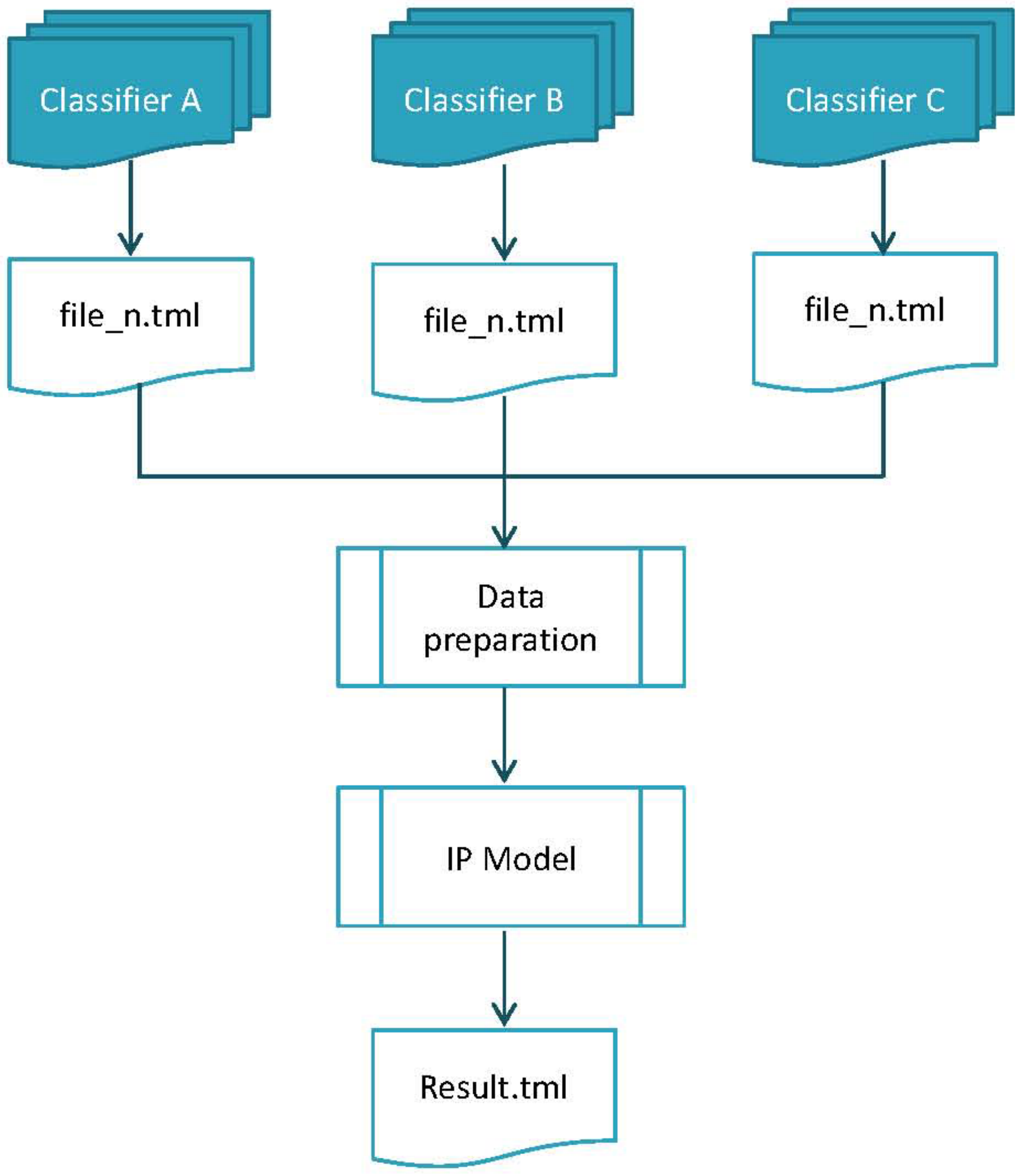

Figure 3-2: Data processing pipeline

The objective of the pipeline is to take data from multiple classifiers, combine the data into an ensemble classifier, optimise the output from the ensemble to assign the most likely class, subject to global rules, and convert the optimised output back to a TimeML-formatted file that can be measured using the TempEval-3 challenge measurement tool. We chose to develop the programs in Python, because it allows rapid development, is easy to learn, has object-oriented capabilities and provides very efficient tools for parsing XML files. Python also allowed us to use Pyomo, an open source software package for modelling and solving mathematical programs in Python.

3.3.1 Data Pre-processing

Input data is provided in the form of XML files, which have already been processed through different classifiers to produce TimeML-formatted XML files. Data pre-processing necessitates the parsing of the TimeML-formatted files so that the relevant data can be provided in a suitable format to the ILP model. The objective function is to maximise the probability of a relationship type ($reltype$) being assigned to a temporal relationship (TLINK). This is formulated as an assignment problem, where the input to the optimiser is a list of TLINKs, each with a set of probabilities representing the probability of each $reltype$.

3.3.2 Input Data Format

A TimeML (.tml) file is an XML file. The XML tags that are relevant to this problem are illustrated in Figure 3-3.

```
<TimeML>
  <DOCID>filename</DOCID>
  <DCT>
    <TIMEX3 tid="t0" type="TIME" value="yyyy-mm-dd"
     attributes</TIMEX3>
  </DCT>
  <TEXT>Newsfeed text goes here
    <EVENT eid="eid" class="eventclass">event_text</EVENT> text
     continues.
    <TIMEX3 tid="tid" type="type" value="value"
           attributes>time_text</TIMEX3>
    Newsfeed text continues with each event tagged with <EVENT> and
    each time expression tagged with <TIMEX3>
  </TEXT>
  <MAKEINSTANCE eventID="eid" eiid="eiid" attributes/>
  <TLINK lid="lid" reltype="reltype" eventInstanceID="eid"
        relatedToEventInstance="eiid">
  <TLINK lid="lid" reltype="reltype" eventInstanceID="eid"
        relatedToTime="tid">
  <TLINK lid="lid" reltype="reltype" timeID="tid"
        relatedToEventInstance="eiid">
  <TLINK lid="lid" reltype="reltype" timeID="tid"
        relatedToTime="tid">
<TimeML>
```

Figure 3-3: TimeML-formatted file

A description of the TimeML tags is taken from (Pustejovsky et al. 2005), with further guidance in (Saurı et al. 2009).

<DCT>	Document Creation time This contains a <TIMEX3> tag whose attribute, *tid* , is normally t0 and whose attribute, *value* , is equivalent to the date of the newsfeed (or date and time).
<TEXT>	This tag surrounds the entire text of the newsfeed.

<EVENT> This tag annotates events that have been identified. Each event is assigned a unique identifier, *eid*, and an event class, *class*.

<TIMEX3> This tag annotates time expressions that have been identified. Each time expression is assigned a unique identifier, *tid*, and other attributes, including the value of the time expression, which could be a date (yyyy-mm-dd), a date range (yyyy-12, for month of December) and other formats.

<MAKEINSTANCE>This tag annotates mappings between events and event instances that will form part of temporal relationships (TLINKs). Generally, there will be a one-to-one relationship between an event (attribute *eid*) and an event instance (attribute *eiid*). However, in a small number of cases, a single event can be linked to more than one event instance. An event instance is created for each instance or realisation of an event. For example, "*Two new flights will leave Bombay on Monday and Tuesday nights from March 30.*" indicates one event (leave) and two event instances (Monday and Tuesday). Each of those event instances may have different temporal relationships with other events/times and so must be split into two instances.

<TLINK> This tag defines the relationship between two entities

- eventInstance-eventInstance
- eventInstance-time interval
- time interval-eventInstance
- time interval-time interval

with one of fourteen possible relationships, attribute *reltype*.

The pseudo-code for the high-level pre-processing task is shown in Figure 3-4.

Pseudo-code for Ensemble processing

```
Ensemble(classifierList, weightingFormula):
for each tml file in classifierList[0]                              [1]
   for each classifier in classifierList                            [2]
         parse events and time expressions
         map event ids and time ids to global ids                   [3]
         convert TLINKs to refer to global ids                      [4]
   end for
   g = getClassifierWeights(weightingFormula)                       [5]
   for each arc in global set
         for each reltype
            assign probabilities using g(x)
         end for
   end for
   store data for optimiser                                         [6]
   create IP model(data)                                            [7]
   get optimised solution
   output new tml file                                              [8]
end for
```

Figure 3-4: Pseudo-code for Ensemble processing

The program is initiated by entering the command:

```
python pipeline.py <dir=classifiers> <file=files>
          <weight=weight> <formula=formula> <opt=optimise>
```

See Running the Ensemble Classifier, for a full description on how to run the program.

[1] There is a separate directory for each of the classifiers and each directory contains the same set of annotated newsfeeds. By default, `dir=CLASSIFIER_` and the program finds all directories that start with this value. Also by default, `file=*.tml` and the program searches for all files matching this description in the directories. The default weight is `weight=1` and the default formula is `formula=1`. See 3.4 for a description of the different weights and weighting formulas.

[2] One file at a time is processed from each of the classifier directories.

[3] EVENT and TIMEX3 tags are mapped to global identifiers so that they can be recognised across all classifiers. For example, to determine whether an annotated event in ClassifierA-file1 is the same as an annotated event in ClassifierB-file1, two conditions must be satisfied:

   i. The actual text corresponding to the event must be identical.
   ii. The starting position of the event text in the full text must be the same.

[4] The TLINKs are stored internally using global identifiers for events/times. The direction of the arc is switched if the right-hand id is alphanumerically less than the left-hand id and the *reltype* is likewise inverted. This ensures consistent representation of arcs across classifiers and is also necessary to implement the transitivity constraint in the optimiser.

[5] After a single newsfeed for each of the classifiers has been parsed, the merged list of TLINKs is scanned and a probability for each of the *reltypes* is calculated. Different methods for calculating these probabilities are discussed in 3.4.

[6] The data to be optimised is stored in Python Dictionary format:

```
{(arc left, arc right) :   (p0, p1, …, p14),
 (arc left, arc right) :   (p0, p1, …, p14),
 (etc.)
}
```

where `arc_left` and `arc_right` are the global identifiers for the two related entities; `p0` to `p14` are the probabilities (0.0 – 1.0) of each of the *relypes*. (`p0` for BEFORE, `p1` for AFTER, etc.) `p14` represents no relationship – NONE – although the probability of this is set to 0.0.

[7] The optimising model is described in 3.5.

[8] The result of the optimiser is converted back to the same Python dictionary format as described in [6] and a new TimeML-formatted tml file is created with the optimised results.

3.3.3 *Mapping events and times across classifiers*

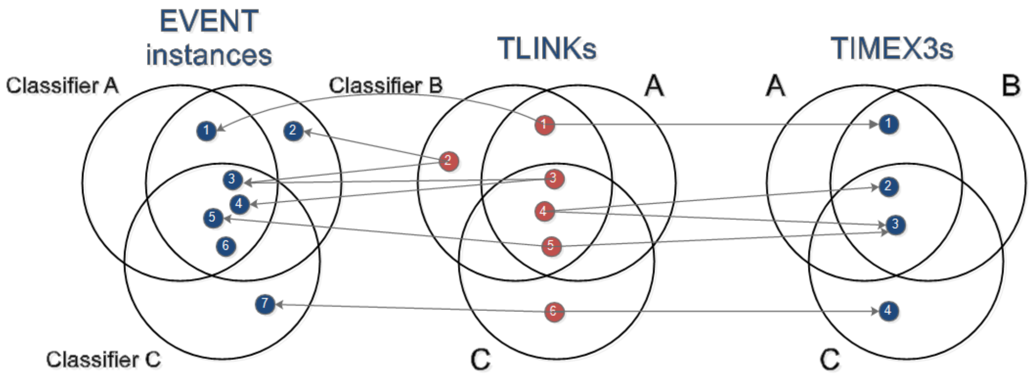

Figure 3-5: Union of TimeML tags across 3 classifiers

Figure 3-5 illustrates the problem of merging annotated events, times and TLINKs across multiple classifiers. Each classifier may annotate a different set of events. Even when the same event is annotated, it may be given a different identifier. Likewise, each classifier may annotate different time expressions or assign different values to the time expressions that are identified. The TLINKs connect events with other events, or events with times, or times with times. Taking the example of a TLINK that connects event instance `ei1` with time expression `t1` in Classifier A, there are several reasons why the same TLINK may not exist in Classifier B.

1. Classifier B does not annotate the event instance `ei1`.
2. Classifier B does not annotate the time expression `t1`.
3. Classifier B annotates the time expression `t1`, but assigns a different value attribute.
4. Classifier B annotates both event and time accurately (possibly with different identifiers), but does not annotate the relationship between them through a TLINK.

Even if Classifier B annotates the same TLINK, it could assign a different relationship type (*reltype*). Figure 3-6 explains how annotated events from different classifiers are mapped to a common global identifier.

Pseudo-code for event mapping

```
For each event in fullText
  Get event_id
  Get event_text and position
    event_indicator = concatenate(position,"_",event_text)
    If event_indicator already in global list of indicators
      Get global_id
    Else
    Increment global_id
      Add event_indicator and global_id to global list of indicators
      Assign global_id to current classifier event
End for
```

Figure 3-6: Pseudo-code for event mapping

3.4 Weighting formulas

One of the objectives is to try different weighting formulas and measure their effect on results. If we consider the ensemble as a panel of experts, it is natural to attach more weight to the opinion of the "best" expert. But how do we define the "best"? In the case of classifiers, we can form opinions on their effectiveness based on scores such as F1, Precision and Recall. A number of different weighting formulas are employed. These are formulated as functions $g^{(n)}(x)$ $n = 1..7$. The input is composed of the following sets and indices:

A	a set of arcs
i	index of arc in A
Y	a set of classifiers
C	set of classes $\{$BEFORE, AFTER, …., NONE $\}$
j	index of class c in C
U_i	the subset of classifiers identifying arc i
W	a set of weights
$w_{k,y}$	weight constant k for classifier y [6]
H_y	the set of predicted classes for classifier y
$h_{y,i}$	the class predicted by classifier y for arc i
$\alpha_{i,j}$	co-efficient of arc i and class j

Table 3-3: Notation used in weighting formulas and IP formulation

Further, we introduce these binary variables:

$$h_{y,i}(c_j) = \begin{cases} 1, \text{ if classifier } y \text{ predicts } c_j \text{ for arc } i \\ 0, \text{ otherwise} \end{cases}$$

The six weighting formulas are as follows:

Formula 1: Total weight divided by number of classifiers detecting arc

$$g^{(1)}(\alpha_{i,j}) = \frac{\sum_{y \in Y} w_{k,y} * (h_{y,i}(c_j) = c_j)}{\sum_{y \in U_i} 1} \qquad (3\text{-}1)$$

Formula 2: Total weight divided by number of classifiers

$$g^{(2)}(\alpha_{i,j}) = \frac{\sum_{y \in Y} w_{k,y} * (h_{y,i}(c_j) = c_j)}{\sum_{y \in Y} 1} \qquad (3\text{-}2)$$

[6] Different weight constants are explained in 3.4.1

Formula 3: Normalised weights with threshold 0.5 on arc probability

$$g^{(3)}(\alpha_{i,j}) = \sum_{y\in Y} \mathrm{norm}(w_{k,y}) * (h_{y,i}(c_j) = c_j) \tag{3-3}$$

$$\mathrm{norm}(w_{k,y}) = \frac{w_{k,y}}{\sum_{y\in Y} w_{k,y}}$$

arc is removed if $g^{(3)}(x_{i,j}) < 0.5$

Formula 4: Sum of logs of probability of classifier being correct

$$g^{(4)}(\alpha_{i,j}) = \sum_{c\in C} \log\left[(h_{y,i}(c_j) = c_j) * w_{k,y} + (h_{y,i}(c_j) \neq c_j) * (1 - w_{k,y}) \right] \tag{3-4}$$

Formula 5: Product of probabilities of classifier being correct

$$g^{(5)}(\alpha_{i,j}) = \prod_{c\in C} \left[(h_{y,i}(c_j) = c_j) * w_{k,y} + (h_{y,i}(c_j) \neq c_j) * (1 - w_{k,y}) \right] \tag{3-5}$$

Formula 6: Inverted loss function

$$g^{(6)}(\alpha_{i,j}) = p(c_j) * \left[1 - \sum_{y\in Y} (h_{y,i}(c_j) \neq c_j) * \mathrm{norm}(w_{k,y}) \right] \tag{3-6}$$

Formula 7: Hinge loss function

$$g^{(7)}(\alpha_{i,j}) = -\left[(1 - p(c_j)) \sum_{y\in Y} (h_{y,i}(c_j) \neq c_j) * \mathrm{norm}(w_{k,y}) \right] \tag{3-7}$$

The – sign is because the IP formulation is Maximise

3.4.1 Weights

There are also different weights that can be used in the formulas:

$w_{1,y}$ = 1 all classifiers have equal weight

$w_{2,y}$ = f1 score for classifier y, measured by temporal_evaluation.py [7]

$w_{3,y}$ = precision score for classifier y, measured by temporal_evaluation.py

$w_{4,y}$ = recall score for classifier y, measured by temporal_evaluation.py

$w_{5,y}$ = f1x score for classifier y, measured by newmetric.py [8]

$w_{6,y}$ = precisionx score for classifier y, measured by newmetric.py

$w_{7,y}$ = recallx score for classifier y, measured by newmetric.py

[7] The temporal_evaluation.py tool was developed for the TempEval-3 challenge (UzZaman et al. 2012)

[8] newmetric.py is a tool developed for this project that measures the accuracy of a classifier in identifying an arc and in classifying correctly identified arcs. See page 57.

3.5 The ILP Model

Following preparation of the data, the dictionary is passed to the optimiser module. This was developed in Pyomo (Python Optimization Modeling Objects), using the Cbc (**C**oin-or **b**ranch and **c**ut) solver. The problem is formulated as a bipartite graph assignment problem $d \circ l \circ g(x)$. We employ the vector of binary variables:

$$x_{i,j} = \begin{cases} 1 \text{ if relationship } j \text{ is assigned to arc } i \\ 0 \text{ otherwise} \end{cases}$$

in the following integer programming formulation:

d is the ILP Objective Function:

Maximise $\sum_i \sum_j l \circ g^{(n)}(\alpha_{i,j}) * x_{i,j}$ (3-8)

s.t.

$$\sum_j x_{i,j} = 1 \quad \forall i \;\; j \in \{1..15\} \tag{3-9}$$

$$x_{pq,a} + x_{qr,b} - \sum_{c \in C'} x_{pr,c} <= 1 \quad \forall pq, qr, pr \;\in A, \quad a, b \in C \tag{3-10}$$

$c \in C' \subset C$, the set of relationships that can be inferred from $a \circ b$

$$x_{i,j} \in \{0,1\}$$

where

$g^{(n)}(\alpha_{i,j})$ is a weight function,[9]

$l(g)$ is a loss function

[9] Weight functions and loss functions are described in 3.4

(3-9) is a mutual exclusivity constraint that guarantees that only one *reltype* can be assigned to each arc (TLINK).

(3-10) performs a transitive consistency check on each triplet of connected arcs in the data. A triplet of connected arcs is such that if event *p* is related to event *q,* and event *q* is related to event *r*, we can infer some type of relation between event *p* and event *r*. For example,

$$p \text{ BEFORE } q \text{ AND } q \text{ BEFORE } r \Rightarrow p \text{ BEFORE } r\,.$$

This is formulated as $x_{pq,a} + x_{qr,b} - \sum_{c \in C'} x_{pr,c} <= 1$. The relation *c* can be one of a number of relations that would satisfy the assignment. The Pyomo model provides a very efficient way of solving the problem.

1. The list of transitive arcs is built in O(n^2) time.
2. Constraints are added for each triplet of connected arcs. There are $k \times 15^2$ possible constraints, where *k* is the number of connected arcs. 15 is the number of *reltypes*, including NONE, which appears in the antecedent of the constraint only and may be assigned as an opt-out *reltype* if all constraints cannot be satisfied.
3. The IP solver ensures consistency of transitivity rules across the *whole* set of arcs.

This approach addresses a number of issues that have been recognised in interval algebra (Allen 1983); namely, constraints are satisfied over subgraphs of three intervals only and not the complete graph (UzZaman et al. 2012); transitive closure on fourteen *reltypes* constitutes an intractable problem and may be solved by a reduction to three relations in point algebra (Vilain and Kautz 1986), or five points (Denis and Muller 2011).

Chapter 4 - Measurements and Results

4.1 The Measurement Process

The objective of the measurement process is to compare the accuracy of an ensemble of classifiers in identifying and classifying temporal relationships with the accuracy of the individual classifiers. We look to the work of UzZaman and Allen, who developed an evaluation metric as part of the TempEval-3 Challenge. Results were measured against a "Platinum" test set, which was manually annotated by experts and previously unseen. The evaluation program produced Recall, Precision and F1 scores for each of the classifiers. A short description of the rationale for scoring is warranted to explain the significance of our results.

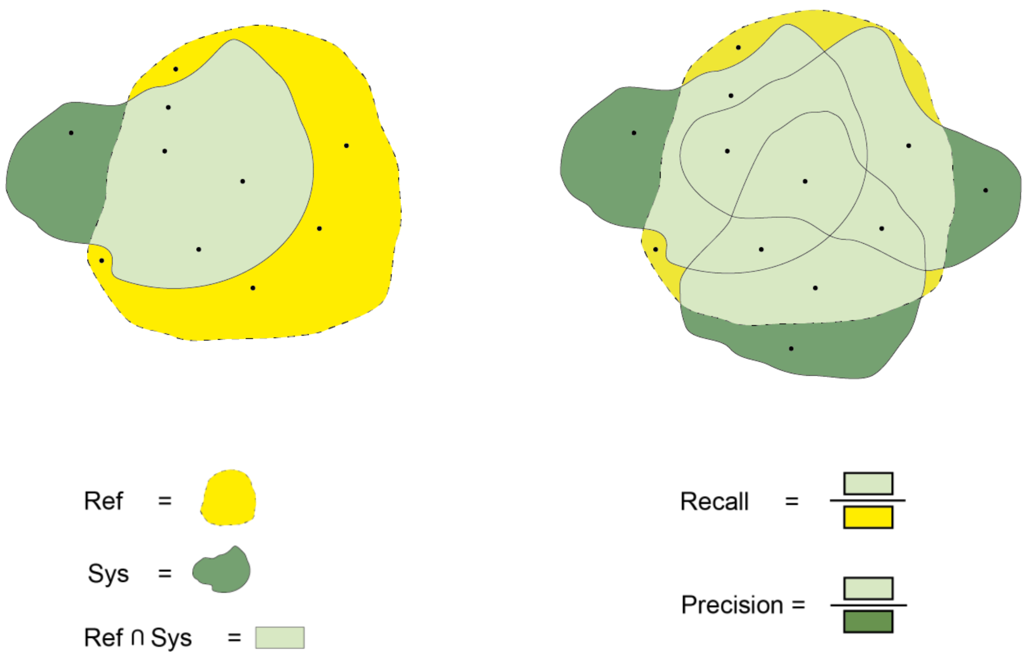

Figure 4-1: How ensembles affect Recall and Precision

A single classifier (green) only covers part of the solution space (yellow). An ensemble of classifiers covers more of the solution space, but may introduce noise (dark green).

$$Recall = \frac{\left| ref^{-}_{relations} \cap sys^{+}_{relations} \right|}{\left| ref^{-}_{relations} \right|} \tag{4-1}$$

sys is the classifier set being evaluated.

ref is the Platinum set.

$set^{-}_{relations}$ is the *reduced* set of TLINKs annotated by the classifier or in Platinum. The *reduced* set excludes redundant relations – i.e. those that can be inferred from other relations.

$set^{+}_{relations}$ is the *closed* set of TLINKs annotated by the classifier or in Platinum, including any additional links that can be inferred from others.

Example of *closed* set and *reduced* set: If the set contains the links A BEFORE B and B BEFORE C, then the *closed* set will also contain A BEFORE C. This ensures that a classifier's Recall score is not penalised for annotating the third relation A BEFORE C, when the Platinum set only annotated the other two relations. If the set contains all three links, then the *reduced* set will remove the third link, as it can be inferred from the other two. This ensures that a classifier's Precision score is not penalised for including the third relation.

Figure 4-1 illustrates the improvement that can be made to the Recall score as a result of using the ensemble of classifiers. $sys^{+}_{relations}$ is likely to be increased, because the number of identified TLINKs is increased. We say "likely" because it is still possible for the addition of a new classifier to cause a TLINK that was previously correctly labelled to be incorrectly labelled. However, this possibility is outweighed by the greater number of TLINKs found. $ref^{-}_{relations}$ remains the same, thus improving the Recall score for the ensemble over individual classifiers.

$$precision = \frac{\left| sys^{-}_{relations} \cap ref^{+}_{relations} \right|}{\left| sys^{-}_{relations} \right|} \tag{4-2}$$

However, the opposite is true for Precision (equation 4.2). $sys^{-}_{relations}$ is increased and consequently, Precision scores can deteriorate due to the number of additional, *irrelevant* TLINKs that are detected.

$$f1 = \frac{\left| sys^{-}_{relations} \cap ref^{+}_{relations} \right|}{\left| sys^{-}_{relations} \right|} \tag{4-3}$$

4.2 The Benchmark

The benchmark data is called the Platinum set. How accurate is the Platinum set itself? This data was manually annotated by experts. To quote Pustejovsky et al,

> *The annotation of TimeML information is on a par with other challenging semantic annotation schemes, like PropBank, RST annotation, etc., where high inter-annotator reliability is crucial but not always achievable without massive preprocessing to reduce the user's workload. In TimeML, inter-annotator agreement for time expressions and events is 0.83 and 0.78 (average of Precision and Recall) respectively, but on TLINKs it is 0.55 (P&R average), due to the large number of event pairs that can be selected for comparison. The time complexity of the human TLINK annotation task is quadratic in the number of events and times in the document.*[10]

The Platinum set was passed through the optimiser to check for global constraint violations and the resulting files compared with the original Platinum set to see if the program and formulation were robust. A number of relationships were changed by the optimiser and these were manually checked. This exercise demonstrated that some relationships in the Platinum set are inconsistent with Allen's rules on transitive closure (Allen 1983). The optimised set was therefore used as an alternative benchmark, although results were measured against both sets.

What is the best score that can be achieved by an ensemble of eight classifiers using the TempEval-3 metric? We analysed the classifiers, ClearTK1-4, Navytime1-2 and UTTime4-5. (For this exercise, we excluded classifiers UTTime-1, UTTime-2 and UTTime-3 as they detect a very high number of links, providing almost complete coverage of the solution space, but at the expense of low precision.)

[10] Pustejovsky et al, 2006: Machine Learning of Temporal Relationships

Number of TLINKs in reduced Platinum set	906
Number of TLINKs in closed ensemble set that are also in reduced Platinum set	821
Number of matching TLINKs in ensemble and reduced Platinum set	472
Maximum Recall score achievable by ensemble	90.62%
Actual Recall score of ensemble	52.10%
Number of TLINKs in closed ensemble set	1,599
Number of TLINKs in closed Platinum set that are also in reduced ensemble	1,088
Number of matching TLINKs in Platinum and reduced ensemble	493
Maximum Precision score achievable by ensemble	68.04%
Actual Precision score of ensemble	30.83%
Maximum F1 score achievable by ensemble 1-2-3-4-5-9-A-B	77.72%
Actual F1 score achieved by ensemble 1-2-3-4-5-9-A-B	38.74%

Table 4-1: Calculating maximum achievable scores for ensemble

The maximum recall score achievable by an ensemble is equivalent to the number of TLINKs that are detected by both the reduced Platinum set AND the closed ensemble set, divided by the reduced Platinum set (821/906 = 90.62%).

The maximum precision score achievable by an ensemble is equivalent to the number of TLINKs that are detected by both the closed Platinum set AND the reduced ensemble set, divided by the reduced ensemble set (1088/1599= 68.04%).

4.3 The Computational Experience

Development and testing proceeded in stages. New insights were formed at each test stage that resulted in the implementation of new weighting formulas and additional pre-processing tasks.

4.3.1 Measuring results from raw newsfeeds

Initial measurements were carried out on TimeML-formatted newsfeeds that had been annotated from raw newfeeds by three classifiers, as part of the TempEval-3 challenge – Task ABC[11]. The weighting formula was a simple proportion of classifiers identifying the arc. See equation (3-1).

Classifier	Baseline	F1	Precision	Recall
Cleartk_2	Rawdata	29.1124	36.2702	24.3141
Cleartk_4	Rawdata	27.3164	32.1244	23.7603
Navytime	Rawdata	20.0778	28.9738	15.3614
Ensemble	Rawdata	26.9265	24.9267	29.2751

Table 4-2: Scores for Task ABC

Table 4-2 indicates the F1, Precision and Recall scores as calculated by the TempEval-3 temporal evaluation program. The scores in all cases are quite low and it might seem that the application of ensemble methods would not be appropriate because none of the classifiers has an individual score greater than 50%. However, if we take a look at the factors that contribute to this score, we can learn how to improve it. The score is a combination of event <EVENT> and time expression <TIMEX3> detection, overlaid with temporal relation detection <TLINK> and classification (*reltype*). If event detection (Task A) and timex detection (Task B) is poor then the annotation of TLINKs (Task C) will be poorer still. A classifier cannot correctly identify and label a TLINK if it did not identify the related events in the first place. See Figure 4-2 for an explanation of how the Platinum solution space might overlap with the solution space for each of the classifiers.

[11] Task A is event detection; Task B is time expression detection; Task C is temporal relation detection and classification.

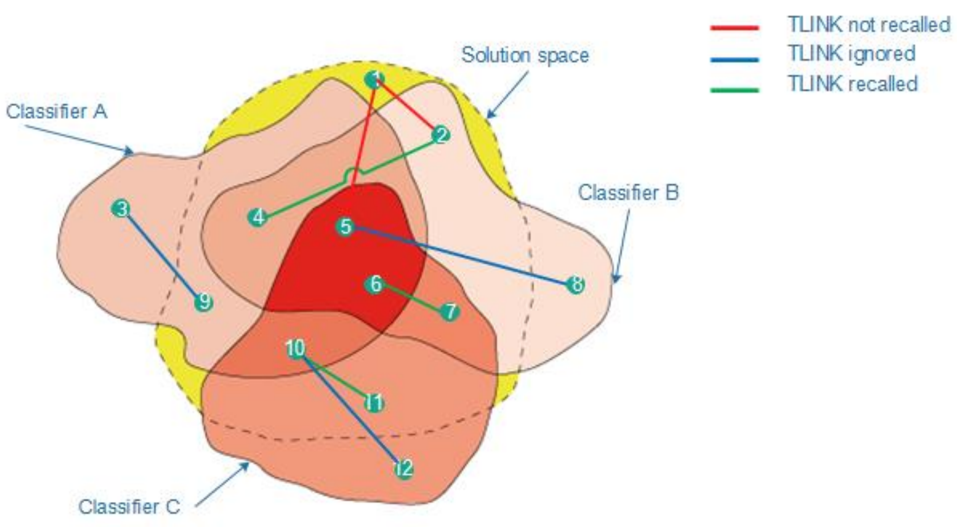

Figure 4-2: Solution space overlap with classifier spaces

In Figure 4-2, we see that the correct TLINKs must be contained within the dotted line. Event 1 is not found in any of the classifiers in the ensemble, so no TLINK involving that event can ever be detected by the ensemble. This reduces the Recall score. Events 3, 8 and 12 are outside the Platinum solution space. TLINKs involving them will be counted, but will be irrelevant and so will affect the Precision score.

4.3.2 Measuring results for TLINK detection and classification only

The scope of this project is to improve TLINK identification and classification, not to improve underlying event detection. Ideally, each classifier will take as input, data that has already been annotated with events and times. Then we are measuring only the TLINK detection and classification. Table 4-3 shows results for Task C only, where the inputs to the classifiers were Platinum-annotated events and times and the classifiers added TLINKs only.

Classifier	F1	Precision	Recall
1-cleartk-1	0.3517	0.3764	0.3300
2-cleartk-2	0.3624	0.3732	0.3521
3-cleartk-3	0.3421	0.3336	0.3510
4-cleartk-4	0.3594	0.3526	0.3664
5-Navytime-1	0.3079	0.3519	0.2737
Ensemble 1-2-3-4	0.3521	0.3408	0.3642
Ensemble 1-2-3-4-5	0.3645	0.3149	0.4327

Table 4-3: Scores for Task C only

Results indicate a similar pattern to Table 4-2. The Recall score is better than the best of the classifiers, but the Precision score is close to the worst. The main cause of the poor precision score is the high number of irrelevant TLINKs that are returned.

4.3.3 Adding a new classifier to the ensemble

We introduced a new classifier – Navytime-1. Even though the F1 score for Navytime-1 was lower than for the other classifiers, we were interested to see whether its inclusion would improve the result. See scores for Ensemble 1-2-3-4-5 in Table 4-3. As expected, the recall score was improved and precision was degraded, but the F1 score showed an improvement over the best classifier. This result supported our theory that adding diverse classifiers to an ensemble improves scores. Task C includes two parts:

1. TLINK detection
2. TLINK classification

Because of the "noise" (irrelevant TLINKs) introduced by each classifier, it was difficult to tell whether we were getting an improvement in the second part of this task. Therefore, we developed a new metric to measure the accuracy of detection (*recallx*) and the accuracy of classification (*precisionx*), with a corresponding *f1x* score.

$$recallx = \frac{|ref_{links} \cap sys_{links}|}{|ref_{links}|} \tag{4-4}$$

$$precisionx = \frac{|ref_{relations} \cap sys_{relations}|}{|ref_{links} \cap sys_{links}|} \tag{4-5}$$

$$f1x = \frac{2 * recallx * precisionx}{recallx + precisionx} \tag{4-6}$$

ref_{links} is the set of arcs (TLINKs) in the Platinum set
sys_{links} is the set of arcs in the ensemble
$ref_{relations}$ is the set of arc relations in the Platinum set
$sys_{relations}$ is the set of arc relations in the ensemble

In this case, *recallx* will measure the number of relevant links (pairs of related entities, excluding *reltype*) that are found by the classifier; and *precisionx* will measure the proportion of those links that are correctly labelled (correct *reltype*).

Table 4-4 illustrates the analysis of *reltype* matches for the ClearTK and Navytime-1 classifiers. We see that in the case of most of the newsfeeds, the ensemble correctly classified more *reltypes* than any of the individual classifiers. However, there were some exceptions to this (highlighted), which raised an interesting question: Why did the optimiser pick an incorrect *reltype*, given that at least one of the classifiers had predicted the correct one? A closer scrutiny revealed the answer, best demonstrated by an example.

In file WSJ_20130321_1145.tml

ei10	INCLUDES	ei15	(Navytime only, Platinum SIMULTANEOUS)
ei15	BEFORE	t0	(all EXCEPT Navytime, link not in Platinum)
ei10	INCLUDES	t0	(all EXCEPT Navytime, Platinum is INCLUDES also)

This violated the transitivity rule,

$$ei10\ INCLUDES\ ei15\ AND\ ei15\ BEFORE\ t0\ \Rightarrow\ ei10\ BEFORE\ t0$$

and so the third relation was changed to BEFORE, even though four out of five classifiers had predicted the correct class for the third relationship, INCLUDES. Given equal weights for each relation in a transitive triplet, the optimiser will always change the consequence of the rule, unless this change violates the transitive rule for another triplet. However, it would have been preferable to change the first relation between ei10 and ei15 to BEFORE.

Perhaps some manipulation of the weights would solve the problem? Weight function $g^{(1)}(x)$ was used for this test – i.e. the weight of each *reltype* was set to the proportion of classifiers that identified the arc (see equation (3-1)). In this example, the first relation INCLUDES had

$$weight = 1/1 = 1$$

We developed weight function $g^{(2)}(x)$ - using proportion of *all* classifiers (see equation (3-2)). So the weight for the first relation was changed to 1/5 = 0.2. Similarly, for the third relation INCLUDES, the weight was changed to 0.8. The result was that when the transitivity constraint was violated, the first relation was changed to IS_INCLUDED and the third relation remained INCLUDES. Overall, results improved significantly. The file WSJ_20130322_804.tml showed an improvement from 11 matching *reltypes* to 16. See Table 4-5.

Note of caution: any change to the weight formula can also result in breaking a *good* TLINK as well as fixing a *bad* one. On average though, the results were better using the weight formula $g^{(2)}(x)$.

filename	ClearTK-1 matched reltypes	ClearTK-2 matched reltypes	ClearTK-3 matched reltypes	ClearTK-4 matched reltypes	Navytime-1 matched reltypes	Ensemble matched reltypes	Platinum links
CNN_20130322_1003.tml	26	26	27	27	25	**37**	102
CNN_20130322_248.tml	4	5	5	5	7	**7**	28
WSJ_20130318_731.tml	7	7	9	8	5	**12**	36
CNN_20130322_314.tml	5	5	5	6	4	**6**	43
nyt_20130321_cyprus.tml	20	21	26	26	9	**28**	80
nyt_20130321_women_senate.tml	18	18	21	21	11	**26**	69
AP_20130322.tml	12	12	12	11	4	**13**	44
WSJ_20130321_1145.tml	**7**	**7**	**9**	**9**	**3**	**7**	**36**
WSJ_20130322_804.tml	**13**	**13**	**14**	**12**	**6**	**11**	**44**
CNN_20130321_821.tml	3	3	2	3	2	**5**	12
nyt_20130321_sarkozy.tml	3	3	3	4	4	**5**	24
nyt_20130321_china_pollution.tml	22	22	23	23	12	**28**	67
bbc_20130322_721.tml	2	4	5	5	2	**5**	30
bbc_20130322_332.tml	13	13	15	15	10	**18**	67
bbc_20130322_1353.tml	6	6	5	5	7	**8**	54
bbc_20130322_1600.tml	2	2	1	2	4	**5**	27
CNN_20130322_1243.tml	**5**	**5**	**6**	**5**	**2**	**5**	**14**
bbc_20130322_1150.tml	24	25	22	26	3	**26**	65
nyt_20130322_strange_computer.tml	4	5	6	6	3	**7**	26
WSJ_20130322_159.tml	6	6	6	6	9	**9**	63

Table 4-4: Analysis of matching reltypes in classifiers and ensemble

filename	ClearTK-1 links	ClearTK-1 matched links	ClearTK-1 matched reltypes	ClearTK-2 links	ClearTK-2 matched links	ClearTK-2 matched reltypes	ClearTK-3 links	ClearTK-3 matched links	ClearTK-3 matched reltypes	ClearTK-4 links	ClearTK-4 matched links	ClearTK-4 matched reltypes	Navytime1 links	Navytime-1 matched links	Navytime-1 matched reltypes	Ensemble links	Ensemble matched links	Ensemble matched reltypes	Platinum links
CNN_20130322_1003	113	29	26	122	29	26	136	31	27	137	31	27	74	30	25	179	49	39	102
CNN_20130322_248	35	4	4	41	5	5	49	7	5	50	7	5	13	8	7	57	11	7	28
WSJ_20130318_731	17	8	7	21	8	7	35	10	9	26	10	8	24	10	5	46	18	11	36
CNN_20130322_314	47	7	5	48	7	5	52	7	5	53	8	6	37	11	4	77	15	6	43
nyt_20130321_cyprus	67	23	20	75	24	21	90	30	26	82	29	26	53	15	9	127	40	29	80
nyt_20130321_women	70	21	18	78	21	18	89	26	21	88	26	21	61	20	11	124	38	26	69
AP_20130322	54	16	12	55	16	12	60	17	12	56	16	11	40	6	4	71	19	13	44
WSJ_20130321_1145	30	9	7	32	9	7	39	14	9	43	16	9	29	12	3	58	21	8	36
WSJ_20130322_804	41	14	13	47	15	13	55	19	14	51	18	12	30	9	6	75	24	16	44
CNN_20130321_821	5	3	3	6	3	3	10	4	2	10	4	3	9	4	2	15	8	5	12
nyt_20130321_sarkozy	27	5	3	29	5	3	40	5	3	38	5	4	18	5	4	48	8	4	24
nyt_20130321_china_po	64	22	22	69	22	22	77	24	23	78	25	23	50	18	12	107	36	29	67
bbc_20130322_721	34	4	2	38	6	4	42	8	5	40	8	5	19	7	2	53	13	5	30
bbc_20130322_332	60	14	13	71	15	13	79	16	15	80	15	15	46	18	10	108	29	17	67

filename	ClearTK-1 links	ClearTK-1 matched links	ClearTK-1 matched reltypes	ClearTK-2 links	ClearTK-2 matched links	ClearTK-2 matched reltypes	ClearTK-3 links	ClearTK-3 matched links	ClearTK-3 matched reltypes	ClearTK-4 links	ClearTK-4 matched links	ClearTK-4 matched reltypes	Navytime1 links	Navytime-1 matched links	Navytime-1 matched reltypes	Ensemble links	Ensemble matched links	Ensemble matched reltypes	Platinum links
bbc_20130322_1353	56	7	6	63	7	6	68	5	5	65	5	5	36	13	7	95	16	8	54
bbc_20130322_1600	30	5	2	32	5	2	41	4	1	44	6	2	25	7	4	65	12	5	27
CNN_20130322_1243	17	6	5	19	6	5	25	6	6	24	6	5	12	4	2	34	9	6	14
bbc_20130322_1150	66	25	24	72	26	25	77	27	22	74	27	26	48	7	3	86	31	25	65
nyt_20130322_str	27	6	4	34	6	5	42	8	6	41	7	6	17	3	3	53	9	7	26
WSJ_20130322_159	75	13	6	88	13	6	87	12	6	83	12	6	50	13	9	126	20	10	63
Totals		241	202		248	208		280	222		281	225		220	132		426	276	931
		RecallX	**.259**		**RecallX**	**.266**		**RecallX**	**.301**		**RecallX**	**.302**		**RecallX**	**.236**		**RecallX**	**.458**	
		PrecisionX	**.838**		**PrecisionX**	**.839**		**PrecisionX**	**.793**		**PrecisionX**	**.801**		**PrecisionX**	**.600**		**PrecisionX**	**.648**	
		F1X	**.396**		**F1X**	**.404**		**F1X**	**.436**		**F1X**	**.438**		**F1X**	**.339**		**F1X**	**.536**	

Table 4-5: Analysis of matching reltypes using weighting formula $g_2(x)$

4.3.4 Introducing the NONE reltype

In tandem with the new measurement tool, we wanted to examine the effect of introducing a new *reltype* – NONE. This was in response to poor Precision scores because too many irrelevant TLINKs were identified by the classifiers. The probability of NONE is set to the proportion of classifiers that do NOT detect the TLINK. The optimiser carries out the assignment of *reltypes* as normal, but now, when NONE is assigned, the TLINK is removed from the final tml file. However, this resulted in a further drop in the F1 score. The expected deterioration in the Recall score was not compensated by an increased Precision score and the F1 score was the lowest yet. See Table 4-6.

Ensemble	F1	Precision	Recall
1-2-3-4	0.3521	0.3408	0.3642
1-2-3-4 with NONE	0.3496	0.3744	0.3278
1-2-3-4-5	0.3645	0.3149	0.4327
1-2-3-4-5 with NONE	0.3483	0.3743	0.3256

Table 4-6: Comparison of results when NONE reltype used

4.3.5 Our Penicillin moment

We decided to abandon the idea of the NONE reltype – at least until we had a sufficient number of classifiers in the ensemble to warrant removal of uncommon TLINKs. The probability of *reltype* NONE was set to 0.0, although it was not completely removed from the data structure. Testing continued and we noticed something strange – the results were better than they were *before* we introduced NONE. Why? We expected them to be the same. Perhaps there was some difference in the data structure that caused the change? We examined one file in detail and established that the data passed to the optimiser was exactly the same in both cases, but in the second case, the solver assigned a value of NONE to one of the arcs. This could only happen if the transitivity rule was violated. Although NONE did not feature at all in the consequence of any of the rules, it did feature in the antecedent. For example,

A BEFORE B and B NONE C => no constraint

We discovered that when there was a chain of linked events, if a transitivity rule was broken, the solver would sometimes break the chain by assigning the *reltype* NONE as this was the lowest cost solution. If the TLINK was irrelevant (as it often was in this scenario), its removal had no effect on Recall, but improved Precision, thus improving the F1 score. NONE is, in effect, a catch-all *reltype* when transitivity constraints cannot be reconciled across the *whole* newsfeed.

4.3.6 Using different weights

We introduced a new parameter to the program – `weight=n`, where `n` has values from 1 to 7 that correspond to different weight coefficients. See a full description of the weights on page 54. The best score achieved was when we applied the PrecisionX score of the classifier as a weight. ClearTK-1 has the highest PrecisionX score and Navytime-1 has the lowest, so the final result was weighted in favour of ClearTK-1's prediction. However, it should be noted that the ensemble, at this stage, was lacking in diversity, because four out of five classifiers were from the ClearTK group.

Ensemble	**Weight**	**F1**	**Precision**	**Recall**
1-2-3-4-5	F1	0.3779	0.3303	0.4415
1-2-3-4-5	Recall	0.3751	0.3267	0.4404
1-2-3-4-5	1	0.3734	0.3266	0.4360
1-2-3-4-5	Precision	0.3716	0.3244	0.4349
1-2-3-4-5	F1X	0.3774	0.3295	0.4415
1-2-3-4-5	**PrecisionX**	**0.3799**	**0.3333**	**0.4415**
1-2-3-4-5	RecallX	0.3774	0.3295	0.4415

Table 4-7: Comparing weights

4.3.7 Adding more diversity

We introduced two more classifiers – UT4 and UT5. UT4 had the lowest F1 score so far, but had the second-highest Precision score. Despite this, its introduction resulted in a further increase in F1 score for the ensemble in all cases, except where the weight used was the F1 score. See Table 4-8 . We then added UT5 and the results were improved even more. Table 4-8 demonstrates that in all cases, the best F1 scores were achieved when the largest number of classifiers was used.

Classifiers	1. Weight = 1 Proportion of classifiers identifying arc			2. Weight = 1 Proportion of all classifiers			3. Weight = classifier F1 score		
	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall
1-2-3-4	0.3521	0.3408	0.3642	0.3532	0.3429	0.3642	0.3595	0.3489	0.3709
1-2-3-4-5	0.3645	0.3149	0.4327	0.3734	0.3266	0.4360	0.3779	0.3303	0.4415
1-2-3-4-5-9	0.3678	0.3058	0.4614	0.3759	0.3126	0.4713	0.3772	0.3170	0.4658
1-2-3-4-5-9-A	0.3762	0.3098	0.4790	0.3850	0.3194	0.4845	0.3900	0.3211	0.4967
Maximum score	**0.3762**			**0.3850**			**0.3900**		

Classifiers	4. Weight = classifier precision score			5. Weight = classifier recall score			6. Weights normalised to sum to 1 and 50% support threshold		
	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall
1-2-3-4	0.3575	0.3480	0.3675	0.3595	0.3489	0.3709	0.3503	0.3518	0.3488
1-2-3-4-5	0.3716	0.3244	0.4349	0.3751	0.3267	0.4404	0.3474	0.3695	0.3278
1-2-3-4-5-9	0.3767	0.3152	0.4680	0.3783	0.3179	0.4669	0.3601	0.3709	0.3499
1-2-3-4-5-9-A	0.3834	0.3158	0.4879	0.3904	0.3220	0.4956	0.3599	0.3823	0.3400
Maximum score	**0.3834**			**0.3904**			**0.3601**		

Table 4-8: Comparison of different weighting formulas

4.3.8 Adding "noisy" classifiers

We introduced classifier UTTime-1. This classifier uses a simple rule-based approach to extract all possible pairs of temporal entities. Unfortunately, the Cbc solver, running on laptops could not cope with the size of the problem. (In one case, there were over 7,000 decision variables and over 360,000 constraints). This problem was solved in seventy-five seconds by the CPLEX solver. The results, as expected, had a detrimental effect on the score because the very high number of TLINKs affected precision.

Ensemble	F1	Precision	Recall
1-2-3-4-5-9-A-6	29.912	20.3819	56.181

4.3.9 Removing TLINKs – a probabilistic method

We developed a new formula to calculate weight, $g^{(3)}(x)$. This formula uses the Precision score of each classifier and normalises their scores so that they sum to 1. TLINKs are only selected for optimisation if

$$g^{(3)}(x_{i,j}) = \sum_{y \in Y} \mathrm{norm}(w_{k,y}) * (h_{y,i}(c_j) = c_j) \geq 0.5$$

This formula resulted in deterioration of the F1 score for most ensembles, primarily due to the lower Recall score, because many TLINKs were omitted from the final tml file. However, the formula was useful when UTTime-1, UTTime-2 and UTTime-3 were included in the ensemble because many TLINKs were removed before optimisation and the IP model was more tractable. See Table 4-9.

Ensemble	F1	Precision	Recall
1-2-3-4	0.3503	0.3518	0.3488
1-2-3-4-5	0.3474	0.3695	0.3278
1-2-3-4-5-9	0.3601	0.3709	0.3499
1-2-3-4-5-9-A	0.3599	0.3823	0.3400
1-2-3-4-5-9-A-6	0.3822	0.3672	0.3985

Table 4-9: Comparing results for formula $g^{(3)}(x)$

4.3.10 Use Bayes formulas to calculate weights

Two more weighting formulas were designed using a naïve-Bayes construct. The formulas utilise the *precisionx* score of the classifier – i.e. the probability of correctly predicting the *reltype*, given the arc exists. The score must be greater than or equal to 0.5 to be compatible with this formula. (See equation (4-5)). Formula $g^{(5)}(x)$ is a product of probabilities and is prone to underflow error as the number of classifiers grows. Formula $g^{(4)}(x)$ uses the sum of the logs to avoid underflow error and also to give a different perspective on the weighting. In Table 4-10, we can see that $g^{(5)}(x)$ gets a better score than $g^{(4)}(x)$ and is the highest F1 score to date for an ensemble.

Ensemble	g(x)	F1	Precision	Recall
1-2-3-4	4	0.3557	0.3455	0.3664
1-2-3-4	5	0.3610	0.3498	0.3731
1-2-3-4-5	4	0.3813	0.3349	0.4426
1-2-3-4-5	5	0.3821	0.3374	0.4404
1-2-3-4-5-9	4	0.3836	0.3219	0.4746
1-2-3-4-5-9	5	0.3848	0.3246	0.4724
1-2-3-4-5-9-A	4	0.3870	0.3188	0.4923
1-2-3-4-5-9-A	5	0.3901	0.3217	0.4956

Table 4-10: Comparison of Bayes formulas

4.3.11 Introducing Loss functions

Up to this point, all of the weighting functions rewarded the probability of making a correct prediction. We wanted to examine alternative formulae that incurred penalties for incorrect predictions, so that we could compare results for minimising the error with those for maximising the probability. Formula (3-6) is an inverted loss function. It ignores correct predictions and looks at the probability of a classifier *incorrectly* predicting a class, and then inverts it through $1-p$ to get the probability of *correctly* predicting the class. Formula (3-7) is a hinge loss function – it also looks at the probability of a classifier *incorrectly* predicting a class. The method is based on the hinge loss formula $\ell(y) = \max(0, 1 - t.y)$ $t.y = 1$ when the hypothesis matches the classifier prediction; $t.y = 0$ otherwise. Therefore, $\ell(y) = 0$ when hypothesis is true or $\ell(y) = 1$ otherwise. $g^{(6)}(x)$ can be solved in the IP model as a Maximise problem. $g^{(7)}(x)$ is a Minimise problem, but this is handled by multiplying the values by -1 before passing to the optimiser where it is solved as a Maximise problem.

Ensemble	g(x)	F1	Precision	Recall	Matched reltypes
1-2-3-4	6	0.3583	0.3485	0.3687	284
1-2-3-4	7	0.3541	0.3389	0.3709	282
1-2-3-4-5	6	0.3768	0.3280	0.4426	346
1-2-3-4-5	7	0.3520	0.3004	0.4249	334
1-2-3-4-5-9	6	0.3835	0.3212	0.4757	371
1-2-3-4-5-9	7	0.3615	0.3005	0.4536	361
1-2-3-4-5-9-A	6	0.3915	0.3216	0.5000	397
1-2-3-4-5-9-A	7	0.3677	0.3005	0.4735	385

Table 4-11: Comparison of loss functions

This table shows that the inverted loss function performs better than the hinge loss function and results in our best F1 score to date of 0.3915, almost three percentage points more than the best classifier, which has a score of 0.3624.

Chapter 5 - Analysis/Discussion

Our objective is to discover whether an ensemble of classifiers can outperform individual classifiers in the task of temporal relationship detection and classification in text. This project entails the development of a tool to experiment on different ensembles, using a variety of weights and weighting formulas and the measurement of results on a sample of newsfeeds.

We endeavour to answer the following questions:

1. Does an ensemble of classifiers achieve a better F1 score than individual classifiers?
2. Is there an optimum ensemble composition?
3. How good is the ensemble at link *classification* as opposed to link *detection*?
4. What is the effect of using different weights and different weighting formulas on the score?
5. Does the IP formulation resolve transitivity violations in the classifiers?
6. How efficient is the IP model in finding a solution?

5.1 Comparing ensembles with individual classifiers

The individual classifier scores are shown in Table 5-1. F1, Precision and Recall scores are calculated using the temporal evaluation tool that was used in the TempEval-3 challenge (UzZaman et al. 2012). See Recall and Precision scoring equations on page 58. Table 5-1 shows that the ClearTK classifiers have considerably higher F1 scores than the other classifiers (apart from UTTime-5), due to their consistent results on Precision and Recall. UT-1, 2 and 3 have the highest Recall scores. This is not surprising as they use a simple rule-based technique to extract as many temporal relations as possible. However, the inclusion of irrelevant links severely hampers the Precision score. The table also demonstrates that there is a steady improvement in F1 scores as each classifier is added to the ensemble, with some small deviations. The main reason for this is the higher Recall scores achieved by ensembles. An ensemble will always *detect* more links than a single classifier. As long as the total number of matched links does not decrease, the Recall score will go up. However, a classifier may introduce a lot of "noise" (irrelevant TLINKs) to an ensemble, so Precision will go down unless the *irrelevant* TLINKs are balanced by improved *classification* of the links.

Classifier	F1	Precision	Recall
1-cleartk-1	0.3517	0.3764	0.3300
2-cleartk-2	0.3624	0.3732	0.3521
3-cleartk-3	0.3421	0.3336	0.3510
4-cleartk-4	0.3594	0.3526	0.3664
5-navytime-1	0.3079	0.3519	0.2737
6-UT-1	0.2428	0.1490	0.6556
7-UT-2	0.2415	0.1487	0.6424
8-UT-3	0.2422	0.1507	0.6170
9-UT-4	0.2882	0.3752	0.2340
A-UT-5	0.3499	0.3605	0.3400
B-navytime-2	0.2568	0.3092	0.2196
1-2-3-4	0.3583	0.3485	0.3687
2-9-5	0.3788	0.3198	0.4647
2-9-5-4	0.3738	0.3122	0.4658
2-9-5-4-A	0.3809	0.3111	0.4912
2-9-5-4-A-B	0.3819	0.3115	0.4934
2-9-5-4-A-B-1	0.3857	0.3148	0.4978
2-9-5-4-A-B-1-3	0.3895	0.3186	0.5011
1-2-3-4-5-9-A	**0.3915**	**0.3216**	**0.5000**

Table 5-1: Individual classifiers and ensembles

5.2 *Composition of ensembles*

Is there an optimum ensemble composition? Ensemble theory suggests that a necessary and sufficient condition for an ensemble of classifiers to be more accurate than any individual members is if the individual constituent classifiers are accurate and diverse (Dietterich 2000). Different ensembles were tested to see if there was an optimal composition. The ensembles were built up gradually with diversity in mind.

(See 3.2 for a description of how classifier diversity was measured). The first ensemble was composed of three classifiers – one from each of the classifier groups, ClearTK, UTTime and Navytime (this ensemble is labelled 2-9-5). Classifiers were then added one at a time from each of the different groups.

Table 5-1 shows that when ensembles were built in the order 2-9-5-4-A-B-1-3, the best result was achieved for the largest ensemble. The scores achieved also support the theory of the importance of diversity. Notice how the ensemble 2-9-5, which is diverse, outperformed 1-2-3-4, which all came from the same stable and had higher F1 scores to start with. It was not possible to test all combinations of ensembles; but it is clear that adding classifiers to the ensemble will improve scores as long as the classifier does not introduce too much "noise". For this reason, we could not include UTTime-1, 2 and 3 for most test scenarios, although there is a place for these classifiers, which will be discussed in 5.4.

5.3 Ensemble performance at classification

How good is the ensemble at link *classification* as opposed to link *detection*? The individual classifier scores vary considerably depending on their approach. The ClearTK classifiers, for example, take a conservative approach to detecting links, resulting in low Recall scores, but protecting their Precision scores. UTTime-1, 2 and 3 take the opposite approach. Although they have a low F1 score according to the TempEval-3 metric, they show much potential because the links they identify give over 90% coverage of the solution space. What we would like to know is, if the link is a valid one, how good are the classifiers and the ensembles at labelling that link? The F1X, RecallX and PrecisionX scores were calculated using a new metric that was designed to measure the effectiveness of the ensemble in detecting and classifying links. (See RecallX and PrecisionX scoring equations on page 63). The RecallX score is a measure of the proportion of Platinum links that the classifier detects. The PrecisionX score is a measure of the proportion of those detected links that are correctly labelled.

Ensemble	F1X	PrecisionX	RecallX	Matches
1-cleartk-1	0.4431	0.8227	0.3032	232
2-cleartk-2	0.4886	0.8179	0.3484	265
3-cleartk-3	0.5157	0.7722	0.3871	278
4-cleartk-4	0.5138	0.7809	0.3828	278
5-navytime-1	0.3938	0.5891	0.2957	162
6-UT-1	0.6977	0.5634	0.9161	480
7-UT-2	0.6914	0.5552	0.9161	473
8-UT-3	0.6932	0.5575	0.9161	475
9-UT-4	0.3756	0.7059	0.2559	168
A-UT-5	0.5033	0.7014	0.3925	256
B-navytime-2	0.3588	0.5078	0.2774	131
2-9-5	0.6130	0.6815	0.5570	353
2-9-5-4	0.6238	0.6753	0.5796	364
2-9-5-4-A	0.6447	0.6684	0.6226	387
2-9-5-4-A-B	0.6471	0.6724	0.6237	390
2-9-5-4-A-B-1	0.6455	0.6690	0.6237	388
2-9-5-4-A-B-1-3	0.6488	0.6724	0.6269	392
2-9-5-4-A-B-1-3-6	0.7189	0.5939	0.9108	503
2-9-5-4-A-B-1-3-6-7	0.7163	0.5903	0.9108	500
2-9-5-4-A-B-1-3-6-7-8	0.7163	0.5903	0.9108	500

Table 5-2: F1X, PrecisionX, RecallX scores for classifiers and ensembles

Table 5-2 shows that as the ensemble grows in size, the number of correctly detected links also grows (RecallX). This is obvious because the ensemble contains the *union* of all TLINKs. PrecisionX fluctuates a little. This is because the measure is a proportion of *detected* links, which is growing as the ensemble grows. Nevertheless, the actual number of correctly labelled TLINKs increases as the ensemble grows, with only one minor deviation when classifier 1 is added. Of the individual classifiers, UTTime-1 scores the highest according to this metric. Ensembles including UTTime-1, 2 and 3 score very highly on this metric. [12] More importantly, when combined with the higher precision of the other classifiers in the ensemble, the accuracy of classification is improved. This demonstrates that ensemble "voting" improves results.

5.4 *Effect of weights and formulas on ensemble performance*

What is the effect of using different weights and different weighting formulas on the score? There is certainly evidence to support the theory that weighting the votes of the classifiers according to previous performance gives better results than giving them equal weight.

Ensemble	g(x)	Weight	F1	Precision	Recall
1-2-3-4-5-9-A	6	PrecisionX	0.3915	0.3216	0.5000
2-9-5-4-A-B	2	Precision	0.3912	0.3195	0.5044
1-2-3-4-5-9-A	6	Recall	0.3910	0.3210	0.5000
2-9-5-4-A-B	2	F1	0.3904	0.3197	0.5011
1-2-3-4-5-9-A	5	PrecisionX	0.3897	0.3230	0.4912
2-9-5-4-A-B-1-3	6	PrecisionX	0.3895	0.3186	0.5011
2-9-5-4-A-B	5	PrecisionX	0.3894	0.3189	0.5000
2-9-5-4-A-B-1	6	RecallX	0.3892	0.3169	0.5044
2-9-5-4-A-B	2	PrecisionX	0.3890	0.3183	0.5000
2-9-5-4-A-B-1-3	5	PrecisionX	0.3890	0.3183	0.5000

Table 5-3: Top 10 scores

An analysis of the top ten scores (Table 5-3) shows that using PrecisionX as the classifier weight is effective. PrecisionX is interesting because it is the only score that is greater than 0.5 for all of the classifiers. Formulas 4 and 5 will only work with weights greater than or equal to 0.5 in any case. A weight of 1 means that in a tie situation, there is a certain randomness to the method of choosing the winning *reltype*. The optimiser will choose. Equal weighting barely made it into the top fifty F1 scores (48th position).

[12] Ensembles including 6, 7 and 8 were created without IP constraints.

Weighting formulas are more difficult to interpret. Certainly, there is evidence to support what does NOT work. Formula $g^{(1)}(x)$, which calculates the weight based on the proportion of classifiers that detected the link, is not effective (43rd position). The problem is that equal weight is given to a *reltype* that was detected by only one classifier, as to a *reltype* that was detected by *all* classifiers. When a transitive rule is broken, the optimiser chooses which relationship to change to fix the inconsistency. Results are clearly better if the optimiser can make a more informed choice.

Formula $g^{(3)}(x)$, which deletes TLINKs that have less than 50% support, was not very effective, except when high-density classifiers were introduced – i.e. classifiers with many irrelevant links.

Formula $g^{(6)}(x)$, which is an inverted loss function, performed very well, getting the highest score of all and occurring in four of the ten top scores. It is interesting that $g^{(7)}(x)$, the hinge loss function, did not rank highly at all. These two formulas are the only ones that use the prior distribution of the *reltype* in the calculation. The consequence of this is that when the set of probabilities for each reltype is calculated, it is highly unlikely that any two *reltypes* will have the same probability. In other formulas, any *reltype* which was *not* predicted by any of the classifiers, will get the same weight. If transitivity rules are broken and the optimiser is forced to choose a reltype that was *not* predicted, the choice is akin to random.

5.5 Effect of IP model on ensemble performance

The IP Model carries out two functions;

1. Maximises the likelihood of a given *reltype*, based on input data that uses a form of Bayesian voting by the classifiers.
2. Enforces adherence to global rules – i.e. only one *reltype* can be assigned to each TLINK; and interval algebra rules are implemented through transitivity constraints.

The model is robust in that it solves the integer algebra rules across the *whole* graph (newsfeed) and returns an optimal solution. When there is a chain of connected tri-node subgraphs, the IP formulation will find a *reltype* that satisfies *all* relations in the chain. The constraint includes the possibility of a NONE relationship and the program will assign this *reltype* to the left-hand-side of the rule if that is the lowest cost option. For example, the following chain of relations, which violate transitivity rules, is presented to the optimiser (weights are underneath):

A BEFORE B BEFORE C BEFORE D BEFORE A

0.5 0.6 0.2 0.8

One solution is to change the last relation D BEFORE A to D AFTER A, but this comes at a high cost of 0.8. The cheapest solution may be to break the chain by assigning the *reltype* NONE to C-D. The optimiser makes the best choice based on the weights given. In Figure 5-1, we see that the two composite relations on the left violate the interval algebra rule:

A INCLUDES B AND B BEFORE C => A BEFORE C

In the top example, the optimiser will change the consequent relationship to BEFORE because the weights are equal. In the lower example, the optimiser will change one of the antecedent relationships to IS_INCLUDED because this minimises the cost.

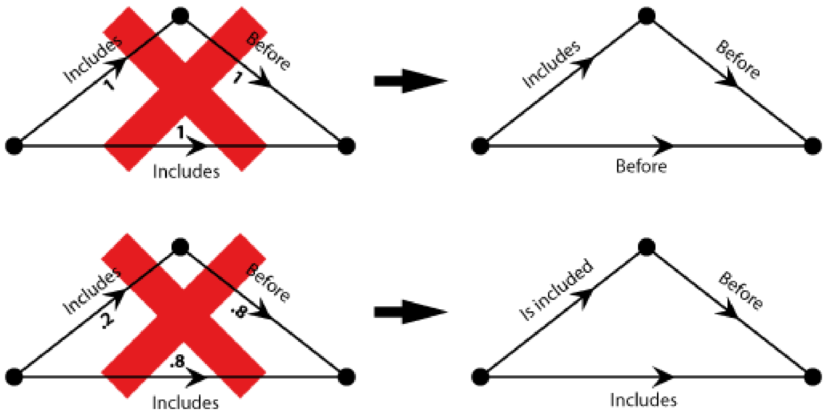

Figure 5-1: Enforcing transitivity constraints

5.6 Performance of IP model

The IP model performed very well. We used an open-source solver Cbc (Coin-or branch and cut), running on laptops. We were able to solve the largest ensembles in seconds for all but one of the files in the test dataset. Integer programming is often criticised for poor performance and IP problems are generally categorised as NP-Hard. Despite not having unimodularity, which would allow us to solve the problem using the LP simplex algorithm, we experienced excellent performance with a basic intuitive assignment formulation. The complexity of the problem was not dictated by the number of decision variables, or by the number of constraints, but by the number of constraint violations in the input data. The IP formulation solves the interval algebra rules over the whole graph. Some other algorithmic approaches satisfy these constraints over subgraphs of three intervals only (UzZaman et al. 2012). We experimented with the addition of “soft constraints” to reduce the problem complexity. The objective is to minimise the number of transitive rule violations rather than eliminate them completely. We did not have sufficient time to develop this idea completely, but it is an area worthy of further development and research.

Table 5-4 gives some data around the complexity of the IP problem. This data relates to our “worst-case” scenario, where we included all eleven classifiers in the ensemble and did not prune any of the TLINKs before optimising. Only one file was not solvable on our laptops running Cbc - WSJ_20130322_159 - but it was solved in 75 seconds on IBM’s CPLEX solver.

Filename	Decision variables	No. of constraints	No. of iterations	Cbc CPU Time	CPLEX CPU time
AP_20130322	2,790	46,434	74	8.01	0.57
bbc_20130322_1150	4,155	86,757	486	23.94	1.02
bbc_20130322_1353	4,440	169,684	5,583	119.10	3.73
bbc_20130322_1600	2,535	76,121	84	16.37	1.49
bbc_20130322_332	3,675	94,621	3	8.56	1.01
bbc_20130322_721	2,115	58,045	0	11.16	0.69
CNN_20130322_1003	7,200	221,568	13,668	181.49	4.69
CNN_20130322_1243	900	12,656	0	0.90	0.14
CNN_20130322_248	2,025	63,679	0	2.00	0.71
CNN_20130322_314	3,045	97,399	902	19.86	1.19
CNN_20130322_821	285	1,523	0	0.04	0.01
nyt_20130321_cyprus	6,285	212,483	207	82.55	4.11
nyt_20130321_china	4,755	162,749	423	55.61	3.76
nyt_20130321_sarkozy	2,130	69,138	78	11.16	2.43
nyt_20130321_women	4,950	132,306	55	41.63	4.04
nyt_20130321_strange	2,820	100,204	1,127	34.82	1.89
WSJ_20130318_731	2,025	48,639	16	5.76	0.43
WSJ_20130321_1145	1,965	41,491	10	6.58	0.64
WSJ_20130322_804	3,270	104,182	5,500	81.87	2.56
WSJ_20130322_159	7,365	367,843	?	3049.83	75.57

Table 5-4: IP Model performance for ensemble of 11 classifiers

Chapter 6 - Conclusions

In drawing conclusions from this research, let us revisit our initial objective – to reconcile the output of multiple stochastic classifiers subject to global rules. Our specific objective was to improve temporal awareness scores, defined as F1, Precision and Recall scores, on a sample set of newsfeeds; to improve the consistency of the output so that it is reliable and useful for inference; and to use ILP methods to ensure this consistency. In this instance, the output of multiple stochastic classifiers is the output from the TempEval-3 challenge with respect to eleven different classifiers and twenty different newsfeeds; the global rules are the temporal reasoning rules as defined by Allen's interval algebra (Allen 1983).

Our methodology was to create an ensemble of classifiers and to implement an Integer Programming formulation to reconcile the output from the classifiers and to enforce global rules. Our aim was to discover the attributes of an ensemble that is better than any of its individual classifiers. Our ensemble used a form of Bayesian Voting and we experimented with different weights and different weighting formulas to see what would give the best outcome.

We can conclude that an ensemble of classifiers is an effective means of improving the temporal awareness scores of Natural Language Processing temporal relations classifiers. The best ensembles are composed of diverse classifiers. In terms of this problem, diversity means that they identify different temporal relations (TLINKs) and they make different class labelling errors. We were able to demonstrate a considerable improvement in F1 score (three percentage points from 36.24 to 39.15, representing an 8% increase on the best classifier score) using an ensemble of eight classifiers from three different classifier groups. There is great potential for further improvements as more classifiers and more diversity are included in the ensemble.

We can also conclude that Integer Programming is an appropriate and effective tool for ensuring compliance with global constraints. The IP formulation performed very well – less than 2 second response time for most samples in ensembles of up to eight classifiers. Moreover, the IP formulation found an optimal solution to the problem and guarantees that the TimeML output, as classified by the ensemble, obeys interval algebra rules.

6.1 Business Contribution

Reliable and consistent temporal relation detection and classification is essential for many applications, such as analysis of newsfeeds and situational reporting. Outside of other uses, this is being relied on by civilian agencies such as FEMA[13], who are responsible for disaster management. Although it may be hard to quantify the value add in terms of dollars, even a small improvement in the coordination of the response to disaster can save numerous lives, which are, indeed, priceless. Our results demonstrated that the ensemble method can improve temporal relation detection and classification, as measured by F1, Precision and Recall scores, by almost three percentage points, and that the resulting annotation is chronologically consistent.

6.2 Academic Contribution

This research is the first example of the application of ensemble methods to solve the problem of temporal relation detection and classification and builds on the excellent work of the TempEval-3 participants. Previous Integer Programming approaches to solving the problem used a reduced set of relationship types. Our IP model is a tractable implementation of transitive closure constraints on the full set of fourteen relationship types as defined by TimeML. We now have a tool to enable experimentation on different weights and weighing formulae to advance the quest for the optimal temporal-labelling classifier.

6.3 Further research

There is endless scope for further research in this area. If we look at the potential of the ensemble to improve scores as described in Table 4-1: *Calculating maximum achievable scores for ensemble*, it is clear that even without the issue of inter-annotator (dis)agreement[14], there is much that an ensemble can achieve in terms of more accurately labelling the classes. Further research on loss functions would be beneficial. Our experience has been that F1 scores are inhibited by poor precision scores as a result of "noise" (irrelevant TLINKs) in the classifiers. The ensemble method can also be further developed to reduce this noise. As the number of classifiers in the ensemble grows, there is room for a probabilistic

[13] Federal Emergency Management Agency

[14] Typically, inter-annotator agreement on temporal relation classification is only 55%

inclusion/exclusion of TLINKs. Another interesting avenue of research is to examine alternative IP formulations. Is there a more efficient formulation that still gives an optimal solution? We only touched on soft constraints, but that is another dimension that requires more investigation.

Appendix A

Running the Ensemble Classifier

A softcopy of all code is provided with this dissertation.

The program is initiated by entering the command:

```
python pipeline.py <dir=classifiers> <file=files>
           <weight=weight> <formula=formula> <opt=optimise>
```

where

`classifiers` = partial name of classifier directories – default “`CLASSIFIER_`”

`filename` = partial name of files to be processed – default “`*.tml`”

`weight` = weight value, default = '1'

'1': same weight across classifiers

'2': F1 score of classifiers – `f1` in scores.dat

'3': Precision score of classifier – `precision` in scores.dat

'4': Recall score of classifiers – `recall` in scores.dat

'5': Extended F1 score of classifiers – `f1x` in scores.dat

'6': Extended Precision score of classifiers – `precisionx` in scores.dat

'7': Extended Recall score of classifiers – `recallx` in scores.dat

`formula` = weighting formula, default = '1'

'1': divide by number of classifiers detecting arc

'2': divide by total number of classifiers

'3': normed weights (sum to 1), 0.5 threshold for arc exclusion

'4': Bayes – sum of logs

'5': Bayes – product of probabilities

'6': Inverse loss function

‘7’: Hinge loss function

`opt` = optimiser options, default = '1'

'0': Do not run optimiser

'1': Do run optimiser

If running the ensemble with weight values '2' to '7', there must be a `scores.dat` file in the same directory as the classifier's `.tml` files. This file holds the different types of weights.

Format of `scores.dat` is:

```
# comments - any number of these starting with #
# following line is comma-separated values
f1,precision,recall,f1x,precisionx,recallx
```

`f1` = F1 score as measured by temporal_evaluation.py

`precision` = precision score as measured by temporal_evaluation.py

`recall` = recall score as measured by temporal_evaluation.py

`f1x` = F1 score as measured by newmetric.py

`precisionx` = precision score as measured by newmetric.py

`recallx` = recall score as measured by newmetric.py

f1x, *precisionx* and *recallx* are probabilities between 0.0 and 1.0. If using weight formulas 4 or 5, the weight must be greater than 0.5.

The file – `reltypes.dat` – resides in the same directory as the program. This file contains the prior probabilities of each *reltype*, calculated from an analysis of the training data.

Appendix B

Classifier Scores

Classifier	F1	Precision	Recall	F1X	PrecisionX	RecallX
1-cleartk-1	0.3517	0.3764	0.3300	0.4431	0.8227	0.3032
2-cleartk-2	0.3624	0.3732	0.3521	0.4886	0.8179	0.3484
3-cleartk-3	0.3421	0.3336	0.3510	0.5157	0.7722	0.3871
4-cleartk-4	0.3594	0.3526	0.3664	0.5138	0.7809	0.3828
5-navytime-1	0.3079	0.3519	0.2737	0.3938	0.5891	0.2957
6-UT-1	0.2428	0.1490	0.6556	0.6977	0.5634	0.9161
7-UT-2	0.2415	0.1487	0.6424	0.6914	0.5552	0.9161
8-UT-3	0.2422	0.1507	0.6170	0.6932	0.5575	0.9161
9-UT-4	0.2882	0.3752	0.2340	0.3756	0.7059	0.2559
A-UT-5	0.3499	0.3605	0.3400	0.5033	0.7014	0.3925
B-navytime-2	0.2568	0.3092	0.2196	0.3588	0.5078	0.2774

Glossary of terms

(compiled using Mathematica and Wikipedia)

algorithm	A step by step procedure for operations in mathematics and computer science
annotation	Data about data (e.g. a comment, explanation, presentational markup) attached to text, image, or other data
arc	An arc of a graph is an ordered pair of adjacent vertices
assignment problem	One of the fundamental combinatorial optimization problems in the branch of optimization or operations research in mathematics. It consists of finding a maximum weight matching in a weighted bipartite graph
Bagging	Bootstrap Aggregating
Bayes Error	In statistical classification, the Bayes error rate is the lowest possible error rate for a given class of classifier
Bayes Optimal Classifier	The Bayes Optimal Classifier is a classification technique. It is an ensemble of all the hypotheses in the hypothesis space. On average, no other ensemble can outperform it, so it is the ideal ensemble
Bayesian voting	An ensemble decision rule that minimizes the posterior expected value of a loss function (i.e., the posterior expected loss)
binary relationship	In mathematics, a binary relation on a set *A* is a collection of ordered pairs of elements of *A*
bi-partite graph	In the mathematical field of graph theory, a bipartite graph (or bigraph) is a graph whose vertices can be divided into two disjoint sets U and V (that is, U and V are each independent sets) such that every edge connects a vertex in U to one in V.
Bootstrap Aggregating	Also called bagging. A machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting
classification	In machine learning and statistics, classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known.
Classifiers	An algorithm that implements classification is known as a classifier.

ClearTK	ClearTK provides a framework for developing statistical natural language processing (NLP) components in Java
Constraint matrix	The set of equations and inequalities defining the set of admissible s olutions in linear programming.
Converses	In logic, the converse of a categorical or implicational statement is the result of reversing its two parts.
convex hull	In mathematics, the **convex hull** of a set *X* of points in the Euclidean plane or Euclidean space is the smallest convex set that contains *X*. For instance, when *X* is a bounded subset of the plane, the convex hull may be visualized as the shape formed by a rubber band stretched around *X*
Coopr	A collection of open-source optimization-related Python packages that supports a diverse set of optimization capabilities for formulating and analyzing optimization models.
corpora	In linguistics, a corpus (plural corpora) or text corpus is a large and structured set of texts (nowadays usually electronically stored and processed).
cost network flow problem	The minimum-cost flow problem looks to find the cheapest possible way of sending a certain amount of flow through a flow network.
CPLEX	IBM ILOG CPLEX Optimization Studio
Decision Theory	Concerned with identifying the values, uncertainties and other issues relevant in a given decision, its rationality, and the resulting optimal decision.
directed graph	In graph theory, a directed graph (or digraph) is a graph, or set of nodes connected by edges, where the edges have a direction associated with them
discriminatio n rule	A rule that assigns x to the group that maximises the prior probability of that classification multiplied by the population density
Disjoint relations	Two relations are disjoint if they have no element in common
Ensemble	Ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms

Eulerian trail	An Eulerian path, also called an Euler chain, Euler trail, Euler walk, or "Eulerian" version of any of these variants, is a walk on the graph edgesof a graph which uses each graph edge in the original graph exactly once.
EVENT	A TimeML tag used to annotate those elements in a text that mark the semantic events described by it
F1 score	A weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0.
Forecasts	**St**atements about events whose actual outcomes (typically) have not yet been observed
Global rules	Constraints applying across other localised constraints for the purpose of ensuring an overall coherent consistency
Gold standard hand-annotation	Standard collection of trustworthy corpora necessary for training and meaningful evaluation of algorithms that use annotations
Greedy search	An algorithm that follows the problem solving heuristic of making the locally optimal choice at each stage in the hope of finding a global optimum
Inference	The act or process of deriving logical conclusions from premises known or assumed to be true. The laws of valid inference are studied in the field of logic.
Integer Programming	A mathematical optimization or feasibility program in which some or all of the variables are restricted to be integers. In many settings the term refers to integer linear programming (ILP), in which the objective function and the constraints (other than the integer constraints) are linear.
Inter-annotator agreement	An Annotation reliability measurement of the proportion of cases on which annotators consistently agree
Linear Programming	LP or linear optimization is a method to achieve the best outcome (such as maximum profit or lowest cost) in a mathematical model whose requirements are represented by linear relationships
Linear programming relaxation	In mathematics, the linear programming relaxation of a 0-1 integer program is the problem that arises by replacing the constraint that each variable must be 0 or 1 by a weaker constraint, that each variable belong to the interval [0,1]

LogiT	Logarithm of the odds. A statistical approach to machine learming. p/(1-p)
machine learning	A subfield of computer science (CS) and artificial intelligence (AI) that deals with the construction and study of systems that can learn from data, rather than follow only explicitly programmed instructions
Machine learning ensemble technique	applying multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms
Markov Logic Networks	A Markov logic network (or MLN) is a probabilistic logic which applies the ideas of a Markov network to first-order logic, enabling uncertain inference. Markov logic networks generalize first-order logic, in the sense that, in a certain limit, all unsatisfiable statements have a probability of zero, and all tautologies have probability one.
Markov Property	The "memoryless" property of a stochastic process.
Markov Random Fields	MRF, Markov network or undirected graphical model is a set of random variables having a Markov property described by an undirected graph.
Mathematica	A computational software program used in many scientific, engineering, mathematical and computing fields, based on symbolic mathematics.
MaxENT	Maxium entropy estimate. A statistical approach to machine learning. Maximally non-committal with regard to missing infomation
multi-graph	A multigraph is a graph that is permitted to have multiple edges , that is, edges that have the same end nodes. Thus two vertices may be connected by more than one edge.
NavyTime	A complete event/time ordering system that annotates raw text with events, times, and the ordering relations between them
NP Hard	A problem is NP-hard if an algorithm for solving it can be translated into one for solving any NP-problem (nondeterministic polynomial time) problem. NP-hard therefore means "at least as hard as any NP-problem," although it might, in fact, be harder.

Optima	A local optimum of an optimization problem is a solution that is optimal (either maximal or minimal) within a neighboring set of candidate solutions. This is in contrast to a global optimum, which is the optimal solution among all possible solutions
path consistency	In constraint satisfaction, local consistency conditions are properties of constraint satisfaction problems related to the consistency of subsets of variables or constraints. Several such conditions exist, the most known being node consistency, arc consistency, and path consistency.
Platinum set	The TempEval-3 Platinum TimeML annotations consists of twenty English newswire documents, each annotated for events, temporal expressions and temporal relations by multiple experts and an adjudicator. This is the corpus used to rank participant systems in the TempEval-3 evaluation exercise. Annotations are in TimeML-strict, a subset of TimeML
Posterior	In Bayesian statistics, the posterior probability of a random event or an uncertain proposition is the conditional probabilitythat is assigned after the relevant evidence or background is taken into account.
Precision	High precision means that an algorithm returned substantially more relevant results than irrelevant
Prediction	In machine learning classification problems the labeling of a tuple
Prior	In Bayesian statistical inference, a prior probability distribution, often called simply the prior, of an uncertain quantity p is the probability distribution that would express one's uncertainty about p before some evidence is taken into account.
Recall	High recall means that an algorithm returned most of the relevant results.
relational database	A relational database is a database that stores information about both the data and how it is related.
scalability	Scalability is the ability of a system, network, or process to handle a growing amount of work in a capable manner or its ability to be enlarged to accommodate that growth
shortest path problem	In graph theory, the shortest path problem is the problem of finding a path between two vertices (or nodes) in a graph such that the sum of the weights of its constituent edges is minimized.
SIGNAL	A TimeML tag representing a temporal signal. These are any function words that suggest a particular temporal relationship.

	Example SIGNALs are: *when, in, after*.
sparseness	A property of many zero values allowing a sparse representation of matrices that do not store every element.
SVM	Non-probabilistic supervised learning binary linear classifier applied to analyse data and detect patterns
TempEval	The aim of TempEval is to advance research on temporal information processing, which could eventually help NLP applications like question answering, textual entailment, summarization, etc.
TimeML	Markup Language for Temporal and Event Expressions
TIMEX3	A TimeML tag primarily used to mark up explicit temporal expressions, such as times, dates, durations, etc
TLINK	TimeML Temporal Link establishing a relationship between two events for the purpose of ordering them in time.
tractability	In computational complexity theory, a problem that can be solved in polynomial time
Transitivity	In mathematics, a binary relation R over a set X is transitive if whenever an element a is related to an element b, and b is in turn related to an element c, then a is also related to c.
transportation problem	The Transportation Problem is a classic Operations Research problem where the objective is to determine the schedule for transporting goods from source to destination in a way that minimizes the shipping cost while satisfying supply and demand constraints
Unimodular	A unimodular matrix is a real square matrix A with determinant det(A)=+/-1
UTTime	A system for annotating temporal relations that uses logistic regression classifiers and exploits features extracted from a deep syntactic parser
Vertex colouring	A vertex coloring is an assignment of labels or colors to each vertex of a graph such that no edge connects two identically colored vertices
Viterbi algorithm	A dynamic programming algorithm for finding the most likely sequence of hidden states – called the Viterbi path – that results in a sequence of observed events, especially in the context of Markov information sources and hidden Markov models

XML	Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

References

Allen, James F. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM* 26 (11):832-843.

Bethard, Steven. 2013. ClearTK-TimeML: A minimalist approach to TempEval 2013. Paper read at Second Joint Conference on Lexical and Computational Semantics (* SEM).

Bethard, Steven, James H Martin, and Sara Klingenstein. 2007. Timelines from text: Identification of syntactic temporal relations. Paper read at Semantic Computing, 2007. ICSC 2007. International Conference on.

Bishop, Christopher M. 2006. *Pattern recognition and machine learning*. Vol. 1: springer New York.

Bramsen, Philip, Pawan Deshpande, Yoong Keok Lee, and Regina Barzilay. 2006. Inducing temporal graphs. Paper read at Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing.

Burke, Edmund K, Jakub Mareček, Andrew J Parkes, and Hana Rudová. 2010. A supernodal formulation of vertex colouring with applications in course timetabling. *Annals of Operations Research* 179 (1):105-130.

Repeated Author. 2012. A branch-and-cut procedure for the Udine course timetabling problem. *Annals of Operations Research* 194 (1):71-87.

Cassidy, Taylor, Bill McDowell, Nathanael Chambers, and Steven Bethard. An Annotation Framework for Dense Event Ordering.

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

Chambers, Nathanael, and Dan Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. Paper read at Proceedings of the Conference on Empirical Methods in Natural Language Processing.

Denis, Pascal, and Philippe Muller. 2011. Predicting globally-coherent temporal structures from texts via endpoint inference and graph decomposition. Paper read at Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three.

Devroye, Luc. 1996. *A probabilistic theory of pattern recognition*. Vol. 31: springer.

Dietterich, Thomas G. 2000. Ensemble methods in machine learning. In *Multiple classifier systems*: Springer.

Do, Quang Xuan, Wei Lu, and Dan Roth. 2012. Joint inference for event timeline construction. Paper read at Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning.

Düntsch, Ivo. 2005. Relation algebras and their application in temporal and spatial reasoning. *Artificial Intelligence Review* 23 (4):315-357.

Granger, Clive William John, and Paul Newbold. 1977. Forecasting economic time series. *Economic Theory and Mathematical Economics*.

Han, J, and M Kamber. 2011. Pei. Data Mining Concepts and Techniques. The Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann Publishers.

Hart, William E, Carl Laird, Jean-Paul Watson, and David L Woodruff. 2012. *Pyomo–Optimization modeling in python*. Vol. 67: Springer.

Ladkin, Peter B. 1990. *Constraint reasoning with intervals: a tutorial, survey and bibliography*: International Computer Science Institute.

Laokulrat, Natsuda, Makoto Miwa, Yoshimasa Tsuruoka, and Takashi Chikayama. 2013. Uttime: Temporal relation classification using deep syntactic features. *Atlanta, Georgia, USA*:88.

Larson, Harold. 1982. Introduction to probability theory and statistical inference.

Mani, Inderjeet, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. 2006. Machine learning of temporal relations. Paper read at Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics.

Punyakanok, Vasin, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. Paper read at Proceedings of the 20th international conference on Computational Linguistics.

Pustejovsky, James, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, and Lisa Ferro. 2003. The timebank corpus. Paper read at Corpus linguistics.

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

Roth, Dan, and Wen-tau Yih. 2002. Probabilistic reasoning for entity & relation recognition. Paper read at Proceedings of the 19th international conference on Computational linguistics-Volume 1.

Repeated Author. 2004. *A linear programming formulation for global inference in natural language tasks*: Defense Technical Information Center.

Repeated Author. 2005. Integer linear programming inference for conditional random fields. Paper read at Proceedings of the 22nd international conference on Machine learning.

Saurı, Roser, Lotus Goldberg, Marc Verhagen, and James Pustejovsky. 2009. Annotating Events in English. TimeML Annotation Guidelines. Brandeis University. Version TempEval-2010.

Talukdar, Partha Pratim, Derry Wijaya, and Tom Mitchell. 2012. Coupled temporal scoping of relational facts. Paper read at Proceedings of the fifth ACM international conference on Web search and data mining.

Tarski, Alfred. 1941. On the calculus of relations. *The Journal of Symbolic Logic* 6 (03):73-89.

Tatu, Marta, and Munirathnam Srikanth. 2008. Experiments with reasoning for temporal relations between events. Paper read at Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1.

UzZaman, Naushad, Hector Llorens, James Allen, Leon Derczynski, Marc Verhagen, and James Pustejovsky. 2012. Tempeval-3: Evaluating events, time expressions, and temporal relations. *arXiv preprint arXiv:1206.5333*.

Verhagen, Marc, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. Semeval-2007 task 15: Tempeval temporal relation identification. Paper read at Proceedings of the 4th International Workshop on Semantic Evaluations.

Verhagen, Marc, Robert Gaizauskas, Frank Schilder, Mark Hepple, Jessica Moszkowicz, and James Pustejovsky. 2009. The TempEval challenge: identifying temporal relations in text. *Language Resources and Evaluation* 43 (2):161-179.

Verhagen, Marc, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. SemEval-2010 task 13: TempEval-2. Paper read at Proceedings of the 5th International Workshop on Semantic Evaluation.

Vilain, Marc B, and Henry A Kautz. 1986. Constraint Propagation Algorithms for Temporal Reasoning. Paper read at AAAI.

Williams, H Paul. 2013. *Model building in mathematical programming*: John Wiley & Sons.

Yoshikawa, Katsumasa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. 2009. Jointly identifying temporal relations with markov logic. Paper read at Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1.