FINAL YEAR PROJECT

A Comprehensive Evaluation of Three Well Recognised OIE Systems

Supervisor: Prof. Roberto Togneri
Student: Jake Weiner
Student number: 20749646
Word Count: 7750

THE UNIVERSITY OF WESTERN AUSTRALIA

Abstract

Information extraction (IE) is the task of extracting structured information from unstructured or semi-structured data, most commonly textual data. A large percentage of IE research is dedicated to what is known as relation extraction – the process of identifying a relationship between two or more entities. This paper evaluates three such relation extraction systems, a new breed of IE system known as open information extraction (OIE). We evaluated three state of the art OIE systems known as ReVerb, OpenIE 4.2 and Stanford OpenIE. These OIE systems were evaluated both in terms of system effectiveness, defined by a F_1 score, and system speed, defined by system runtime in CPU seconds. The PENN-100 dataset was chosen as the dataset input for the three systems. A novel extraction evaluation process is presented in this report. Our findings showed that ReVerb was the fastest system whilst OpenIE 4.2 displayed the greatest system effectiveness. Stanford OpenIE was shown to be the slowest system by a considerable margin, additionally achieving a F_1 score similar to OpenIE 4.2.

Table of Contents

2. Introduction

Information Extraction (IE) systems are becoming ever more prevalent within the Natural Language Processing (NLP) field, gaining notoriety in tasks such as Question Answering [8] and Information Retrieval [9]. The newest breed of IE systems are known as Open Information Extraction Systems (OIE). Whilst traditional IE systems rely on hand-crafted patterns, specific to the domain of data to be analysed, OIE systems rely on textual parsing and machine learning to extract meaningful information. This method of analysis translates to OIE systems having the ability to operate on domain independent data, as well as being able to process large datasets. Two of the most recent OIE system releases are OpenIE 4.2 [17] and Stanford OpenIE [5], both systems authored by teams well respected in the NLP research field. OpenIE 4.2 is the latest OIE system released in the KnowItAll project, created by a team from the University of Washington (UW). OpenIE 4.2 was updated as recently as June 2016. Stanford OpenIE is an OIE system developed by the Stanford NLP group, the same group responsible for creating other NLP programs such as the Stanford co-reference resolution system and the Stanford dependency parser. Stanford OpenIE was developed in 2015. Currently the only paper describing any results from these two OIE systems is [2], for which F_1 scores (a common evaluation metric amongst NLP systems) are given on the 2013 TAC Knowledge Based Population (KBP) slot-filling task. No experiments have yet been performed on these two systems to evaluate system speed, another important OIE system performance metric. It is also worth noting that the tests carried out in the 2013 KBP slot-filling challenge were on a previous version of OpenIE, OpenIE 4.0.

ReVerb [18] is another OIE system developed by the team from the UW, and has gained significant recognition in the OIE research field. ReVerb was developed in 2011 and unlike both OpenIE 4.2 and Stanford OpenIE, does not implement a dependency parser. ReVerb is considered a first generation OIE system, relying on shallow parsing in the form of a Part of Speech (POS) tagger and a Named Entity Recognition (NER) program. Whilst ReVerb has been evaluated previously, it is used a benchmark for system performance in this report.

This report provides a comprehensive evaluation of all three OIE systems, ReVerb, OpenIE 4.2 and Stanford OpenIE. This report is the first to provide system runtime measurements for both OpenIE 4.2 and Stanford OpenIE. This report is also the first to present results for the latest version of OpenIE, OpenIE 4.2. Lastly, this report presents a novel OIE system extraction evaluation method.

The structure of this report is as follows:

In section 3 we provide background on OIE systems, specifically the three systems explored in this report. This background is presented in the form of a literature review. Section 4 discusses the methodology we adopted to evaluate the three OIE systems. Section 5 presents the findings of the system tests carried out. Section 6 discusses the insights these findings provide. Section 7 provides a conclusion of this report and offers suggestions for potential areas of future work.

3. Background

The following section is split into three sub-sections. The first subsection examines the evolution of Information Extraction (IE) systems and details the creation of Open Information Extraction (OIE) systems. This subsection also describes how OIE systems are traditionally evaluated performance wise.

The second subsection contains a detailed review on the three OIE systems investigated in this report. This review contains information published by the authors of the respective systems. The review analyses each system with regards to the following key aspects:

1. The theory behind each system's operation
2. The methodology of how each system has been evaluated previously
3. The results obtained from each system's previous tests

The final subsection describes a previous paper also attempting to evaluate different OIE systems. This section describes the dataset selection choice and the system extraction evaluation methodology implemented in that paper.

3.1 Evolution of IE systems

Information extraction is the task of obtaining meaningful information from an unstructured data source [10]. Information extraction is a rapidly growing field within artificial intelligence research, and has many applications including; question answering, text summation, text similarity and knowledge population tasks [1].

3.1.1 OIE System Creation

Traditional IE systems operated on hand-labelled data or handcrafted patterns and therefore were domain specific [3]. Traditional systems often operated on domain specific linguistic processing, making them reliant on the language used within that domain (e.g news, finance or medicine). In addition to being domain dependant, previous IE systems were also unable to scale to accommodate large dataset analysis, often running in $O(n^2)$ time [3]. A team from the University of Washington (UW) created a new type of IE system, known as Open Information Extraction (OIE), which was a new breed of IE system. OIE systems were designed to operate on both heterogeneous corpora of data, as well as large corpora of data. The team from UW created the first OIE system, known as TextRunner, which was able to run up to 80 times faster than previous IE systems [3].

3.1.2 OIE Generation One

The first generation of OIE systems relied on shallow text analysis, often implementing other Natural Language Processing (NLP) techniques such as

Named Entity Recognition (NER) and Part of Speech (POS) tagging. Examples of NER and POS tagging are shown in figures 1 and 2 respectively.

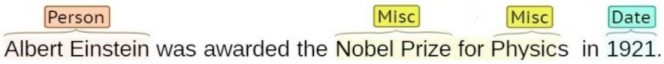

Figure 1 – NER example [1]

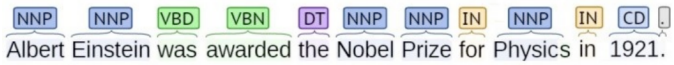

Figure 2 – POS tagging example [1]

NER is the task associated with identifying entities, such as persons, locations, dates etc… in a sentence. POS tagging annotates a sentence, labelling words as nouns, adverbs, verbs, pronouns etc….

Systems such as TextTunner [3], WOE^{pos} [11], R2A2 [12] and ReVerb [18], rely on shallow parsing such as NER and POS tagging. The main benefit of this shallow analysis is increased system speed and simpler OIE system designs. These systems are able to scan large data corpora in relatively fast runtimes (compared to second generation OIE systems). The trade-off for this system speed increase is often a reduction in the number of relations able to be found. The shallow NLP techniques often miss a large number of relations that may exist within the dataset [3][7].

3.1.3 OIE Generation Two

The second generation of OIE systems often include dependency parsers to pre-process textual input. OIE systems that run on this analysis method include Wanderlust [1], WOE^{parse} [13], KrakeN [14], OLLIE [15], ClauseIE [16], OpenIE 4.2 [17] and Stanford OpenIE [5]. As a result of using a dependency parser, these systems are able to extract a larger number of relations from datasets, as well as achieve higher rates of extraction correctness [3]. The trade-off for this deeper textual analysis is often an increase in system runtime [7][16]. An example of a dependency parser analysis is shown in figure 3.

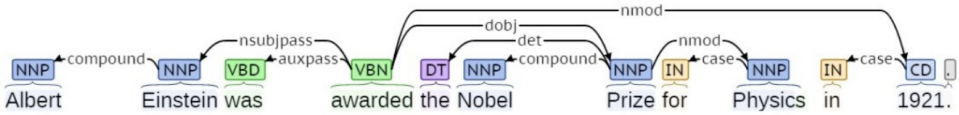

Figure 3 – Dependency parser [1]

A dependency parser creates a tree structure labelled a parse tree. The tree structure shows the relationships between different words, describing which words are “head words” and which words are modifiers [23].

3.1.4 System Evaluation Metrics

3.1.4.1 System Effectiveness

As with many other NLP systems, some common analytic metrics exist to evaluate OIE systems. These metrics form what is known as the System Effectiveness (SE) [7]. There are two components of SE, precision and recall.

Precision is the percentage of correct relations that the system outputs. Precision is defined as [18]:

$$Precision = \frac{no.\ of\ correct\ relations\ system\ found}{total\ no.\ of\ relations\ system\ found}$$

Recall is the percentage of relations that the system is able to find within the dataset. Recall is defined as [18]:

$$Recall = \frac{no.\ of\ correct\ relations\ system\ found}{total\ no.\ of\ relations\ in\ dataset}$$

Precision and recall are most often inversely proportional to each other. Generally, a system is either able to produce high precision but low recall, or low precision with high recall. The trade-off occurs as a result of systems either extracting fewer relations, but with a higher proportion being correct, or a large number of extractions with a smaller percentage of the relations found being correct. It is because of this inverse nature that a weighted average of precision and recall is used. This weighted average is called a F_1 score and is defined as [18]:

$$F_1 = 2.\frac{Recall\ .\ Precision}{Recall\ +\ Precision}$$

3.1.4.2 System Speed

A second, less common metric to evaluate OIE systems is the system speed. The system speed is often defined in terms of sentences analysed per second [3][18]. This metric is often only reported in older OIE systems, such as TextRunner as ReVerb.

3.2 ReVerb

Analysis of this system is based on [18].

3.2.1 System mechanics

Background

ReVerb is the second OIE system developed by the Universty of Washington (UW) team as part of the KnowItAll project. ReVerb has been cited by 576 other research projects according to google scholar, making it one of the most well recognized OIE systems available.

Extraction

ReVerb implements the OpenNLP POS tagger and NP-chunker [21]. ReVerb implements two constraints in its' extraction process. The first constraint is a syntactical one. The syntactical constraint introduced in ReVerb is shown in figure 4.

$$V \mid VP \mid VW^*P$$
$$V = \text{verb particle? adv?}$$
$$W = (\text{noun} \mid \text{adj} \mid \text{adv} \mid \text{pron} \mid \text{det})$$
$$P = (\text{prep} \mid \text{particle} \mid \text{inf. marker})$$

Figure 4 – Syntactical constraints of extractions found [18]

Textual data is input into the POS tagger and NP-chunker and output sentences are scanned for verbs. For each verb, the longest sequence of words satisfying the syntactical constraint shown in figure 4 is extracted. Therefore only relations which are verb-mediated are extracted by ReVerb.

A lexical constraint is then used to filter out over-specified extractions. Some extractions may satisfy this syntactical constraint but fail to represent an actual relation. An example of this is the sentence

"*The Obama administration is offering only modest greenhouse gas reduction targets at the conference*."

which may produce the tuple

(Obama administration, is offering only modest greenhouse gas reduction targets at, the conference)

which is not a valid relation between the "Obama administration" and "the conference". To implement this lexical constraint, extractions from over 500 million web sentences were produced such that all extractions satisfy the syntactic constraint. Heuristically, relation tuples were then extracted from this subset by looking for the pattern

(noun phrase, relation phrase, noun phrase)

Only relation phrases that were compatible with at least 20 different argument pairs were then extracted from these tuples. These relations were then stored in a dictionary to use as a lexical check for future extracted tuples.

Learning

Finally, extracted tuples that satisfy both the syntactical and lexical constraints, are passed to a logistic regression classifier which was trained with 1,000 different hand labelled extractions.

Weight	Feature
1.16	(x, r, y) covers all words in s
0.50	The last preposition in r is *for*
0.49	The last preposition in r is *on*
0.46	The last preposition in r is *of*
0.43	$len(s) \leq 10$ words
0.43	There is a WH-word to the left of r
0.42	r matches VW*P from Figure 1
0.39	The last preposition in r is *to*
0.25	The last preposition in r is *in*
0.23	10 words $< len(s) \leq 20$ words
0.21	s begins with x
0.16	y is a proper noun
0.01	x is a proper noun
-0.30	There is an NP to the left of x in s
-0.43	20 words $< len(s)$
-0.61	r matches V from Figure 1
-0.65	There is a preposition to the left of x in s
-0.81	There is an NP to the right of y in s
-0.93	Coord. conjunction to the left of r in s

Figure 5 - Features used to train logistic regression classifier [18]

This classifier assigns confidence scores to each extraction. The confidence scores allow Area Under the Curve (AUC) plots to be generated. These plots sweep through different confidence levels with precision vs recall being plotted on the axes.

3.2.2 Previous Experiments

The dataset used to perform initial tests on the ReVerb system consisted of 500 sentences from the web, generated via Yahoo's random link service.

Two human judges examined each extraction produced from ReVerb and four other OIE systems, to validate whether or not each extraction was correct or incorrect. The judges agreed upon 86% of the extractions that were found. It was these agreed upon system extractions that defined the total number of relations within the dataset.

No single F_1 score was produced, instead an AUC plot was generated for different confidence thresholds. As this metric is not used to evaluate either OpenIE 4.2 or Stanford OpenIE, this metric is not relevant to this report.

3.3 OpenIE 4.2

Analysis of this system is based on [17].

3.3.1 System Mechanics

Background

The latest OIE system to be released from the UW team is known as OpenIE 4.2. Whilst no paper has been published to describe the theory and tests done on this system, the team have revealed on their "Open IE by knowitall" github page[17], that it is based on a variety of systems. These systems include the older systems such as ReVerb, as well as newer systems SRLIE and Relnoun.

SRLIE and Relnoun

SRLIE addresses the previous problems in OIE systems in that n-ary extractions were not possible. N-ary extractions are relationships between two or more entities. In the previous systems only binary relations were able to be found in the form of

(Entity 1, Relation, Entity 2)

leaving all n-ary relations remaining undiscovered. SRLIE uses a dependency parser and runs the graph of the parser through a Semantic Role Labelling (SRL) system provided by NLP Research group at Emory University. The output of this SRL system are SRL Frames which are reintroduced into the SRLIE system to produce n-ary frames.

Relnoun is an extractor used for noun-mediated relations. The Relnoun github page provides an example of such a relation

(United States, president, Barack Obama)

where the noun, *president,* is the relation between the two entities, United States and Barack Obama.

As a result, OpenIE 4.2 is not only able to produce verb-mediated relations as described in ReVerb, but also noun-medatied relations.

3.3.2 Previous Experimentation

In [5], the team from Stanford responsible for Stanford OpenIE, reported results from the annual Textual Analysis Conference (TAC) Knowledge Based Population (KBP) slot-filling task. The task consisted of filling in attribute slots for different entities within a dataset. In [5] it is reported that the F_1 Score achieved by a previous version of OpenIE 4.2, OpenIE 4.0, was 0.196 in the slot filling

challenge. OpenIE 4.0 achieved a precision score of 0.698 and a recall score of 0.114.

The extraction evaluation methodology employed in the KBP slot-filling task is different from the method described previously in section 3.2.1. The KBP slot-filling task consists of finding pre-defined relationship types that exist between pre-defined entities, in essence leaving many relations undiscovered. Shown in figure 6 are the pre-defined entity types and relations to be filled in by the OIE systems tested.

Person Slots			Organization Slots		
Name	**Type**	**List?**	**Name**	**Type**	**List?**
per:alternate_names	Name	Yes	org:alternate_names	Name	Yes
per:date_of_birth	Value		org:political_religious_affiliation	Name	Yes
per:age	Value		org:top_members_employees	Name	Yes
per:country_of_birth	Name		org:number_of_employees_members	Value	
per:stateorprovince_of_birth	Name		org:members	Name	Yes
per:city_of_birth	Name		org:member_of	Name	Yes
per:origin	Name	Yes	org:subsidiaries	Name	Yes
per:date_of_death	Value		org:parents	Name	Yes
per:country_of_death	Name		org:founded_by	Name	Yes
per:stateorprovince_of_death	Name		org:date_founded	Value	
per:city_of_death	Name		org:date_dissolved	Value	
per:cause_of_death	String		org:country_of_headquarters	Name	
per:countries_of_residence	Name	Yes	org:stateorprovince_of_headquarters	Name	
per:statesorprovinces_of_residence	Name	Yes	org:city_of_headquarters	Name	
per:cities_of_residence	Name	Yes	org:shareholders	Name	Yes
per:schools_attended	Name	Yes	org:website	String	
per:title	String	Yes			
per:employee_or_member_of	Name	Yes			
per:religion	String	Yes			
per:spouse	Name	Yes			
per:children	Name	Yes			
per:parents	Name	Yes			
per:siblings	Name	Yes			
per:other_family	Name	Yes			
per:charges	String	Yes			

Figure 6 – The attribute slots to be filled in the 2013 KBP slot-filling task [6]

Additionally, the system effectiveness scores in the KBP slot-filling task were calculated via an automated evaluation process, the mechanics of which are not described in detail by the TAC. This automatic evaluation process does not include human checking, introducing the potential for evaluation errors to go unnoticed.

In this project we tested the latest version of OpenIE, OpenIE 4.2, to obtain more up to date system effectiveness scores. In addition, the system extractions were manually checked against an annotated dataset to ensure the accuracy of all extractions produced.

Another difference in system evaluation methodology between [18] and the KBP slot-filling task is that no system runtimes were recored for the KBP slot-filling task. This is another novel test that was carried out in this project

3.4 Stanford OpenIE

Analysis of this system is based on [5].

3.4.1 System Mechanics

Background

Where previous systems such as ReVerb and TextRunner rely on a shallow syntactical sweep to identify patterns such as:

(noun, verb, noun)

the Stanford OpenIE system takes a different approach and uses a dependency parser like OpenIE 4.2. The Stanford OpenIE system attempts to split an input sentence into smaller utterances using a multinomial logistic regression classifier.

Learning

The features used to train this classifier are shown in figure 7, where feature class data is extracted via a dependency parser.

Feature Class	Feature Templates
Edge taken	$\{l, \text{short\_name}(l)\}$
Last edge taken	$\{\text{incoming\_edge}(p)\}$
Neighbors of parent	$\{\text{nbr}(p), (p, \text{nbr}(p))\}$
Grandchild edges	$\{\text{out\_edge}(c), (e, \text{out\_edge}(c))\}$
Grandchild count	$\{\text{count}(\text{nbr}(e_{\text{child}})) (e, \text{count}(\text{nbr}(e_{\text{child}})))\}$
Has subject/object	$\forall_{e \in \{e, e_{\text{child}}\}} \forall_{l \in \{subj, obj\}} \mathbb{1}(l \in \text{nbr}(e))$
POS tag signature	$\{\text{pos}(p), \text{pos}(c), (\text{pos}(p), \text{pos}(c))\}$
Features at root	$\{\mathbb{1}(p = root), \text{POS}(p)\}$

Figure 7 - Features used in the multinomial logistic regression classifier [5]

Once a sentence has been split into smaller utterances via the classifier, natural logic semantics are applied to further reduce this set. By breaking up the sentences into smaller utterances, extraction is now a much easier process.

Extraction

The utterances are then checked against 14 hand-crafted dependency parse patterns to extract tuples, as shown in figure 8.

Input	Extraction
cats play with yarn	(cats; play with; yarn)
fish like to swim	(fish; like to; swim)
cats have tails	(cats; have; tails)
cats are cute	(cats; are; cute)
Tom and Jerry are fighting	(Tom; fighting; Jerry)
There are cats with tails	(cats; have; tails)

Input	Extraction
Durin, son of Thorin	(Durin; is son of; Thorin)
Thorin's son, Durin	(Thorin; 's son; Durin)
IBM CEO Rometty	(Rometty; is CEO of; IBM)
President Obama	(Obama; is; President)
Fischer of Austria	(Fischer; is of; Austria)
IBM's research group	(IBM; 's; research group)
US president Obama	(Obama; president of; US)
Our president, Obama,	(Our president; be; Obama)

Figure 8 – Hand-crafted dependency parse patterns [5]

3.4.2 Previous Experiments

As mentioned previously in section 3.3.2, the Stanford OpenIE was evaluated on the 2013 TAC KPB slot-filling challenge. Shown in figure 9 are the results from the slot-filling challenge.

System	P	R	F_1
UW Official*	**69.8**	11.4	19.6
Ollie†	57.4	4.8	8.9
+ Nominal Rels*	57.7	11.8	19.6
Our System			
- Nominal Rels†	64.3	8.6	15.2
+ Nominal Rels*	61.9	13.9	22.7
+ Alt. Name	57.8	17.8	27.1
+ Alt. Name + Website	58.6	**18.6**	**28.3**

Figure 9 – Results from the 2013 TAC KBP slot-filling challenge [5]. Results are given as percentages.

The comparable systems are the UW Official, which was the OpenIE 4.0 system, and the "Our System + Nominal Rels", which was Stanford OpenIE. From these results it can be seen that the Stanford OpenIE system performs better in terms of system effectiveness, displaying a greater F_1 score than OpenIE 4.0.

In total 42,662,862 extractions were produced from analysis of the corpus provided.

There are no reports of any Stanford OpenIE speed tests, and this report attempts to be the first to do so. Additionally no mention of any other experiments on Stanford OpenIE have been reported on since the KBP slot-filling task. This report is the first to evaluate the Stanford OpenIE system based on a methodology different to that used in the KBP slot-filling task.

3.5 Previous OIE System Evaluation

A different OIE system evaluation method than the methods implemented in [18] and [5] was proposed in [2] by a team from the University of Alberta. This new method involved manually annotating a dataset, ranging from 100 to 500 sentences, with a relation (termed a "trigger") and the two entities involved. Not all relations that exist within the dataset were annotated. Instead, the relations that were annotated formed the total number of relations and it was this value that was used in the precision and recall calculations. The system extraction evaluation process consisted of finding a relational word between two annotated entities, via an allowed series of tokens. If the OIE systems tested we able to find the relational word, as well as the entities involved, the extraction was deemed correct.

4. Methodology

The evaluation of the three OIE systems investigated in this report consisted of several phases.

The first phase was to evaluate the three OIE systems based on System Effectiveness (SE). SE, as described previously, is a measure of how many relations the systems are able to find in the dataset (recall), and what percentage of these system extractions are correct (precision). This first phase involved identifying which SE evaluation technique would be used in this report. The three SE evaluation approaches discussed in section 3 were assessed to determine which approach, if any, was the most suitable. The next step was a selecting an appropriate dataset. This selection process was reliant on the SE approach chosen previously. Finally a system extraction evaluation program was created to implement the SE evaluation technique and dataset selected.

The second phase of testing focused on system speed analysis. This phase involved creating several different sized datasets for system testing, recording runtimes for all experiments carried out.

4.1 Previous SE Evaluation Approaches

4.1.1 Approach One – System extracts

As mentioned previously in section 3.2.3, ReVerb was evaluated in conjunction with other OIE systems when originally tested in [18]. In [18], the authors ran ReVerb as well as TextRunner, Woe^{pos} and $\text{Woe}^{\text{parse,}}$ on a dataset of 500 sentences, obtained via Yahoo's random link service. Two human judges then deemed the system extractions,

(Entity 1, Relation, Entity 2)

as either correct or incorrect, eventually agreeing upon 86% of these extractions. These agreed upon extractions were used to define the total number of relations that existed within the dataset.

4.1.1.1 Advantages

Using different OIE systems to determine the total number of relations within a dataset reduces the amount of human input associated with this task. Manual annotation of a dataset is prone to human error and is dependent on the linguistic skills of the annotator.

Whilst all relations were produced via the OIE systems tested, there still exists some form of human examination. The manual checking process carried out by

the two human judges instils greater confidence in the extractions produced by the systems.

4.1.1.2 Disadvantages

When relying on the OIE systems tested to determine the total number of relations that exist within a dataset, any relations not found by the systems remain undiscovered. System flaws, prevalent especially amongst earlier OIE systems such as those tested in this experiment, may result in a substantial amount of relations remaining unfound. This can affect the recall values calculated for each system tested.

Limiting the dataset size is also a consequence when approach one is taken. Large datasets, and the extractions that ensue, would be unfeasible to manually inspect. One such example is the KBP slot-filling task mentioned previously, in which the Stanford OpenIE system produced 42,662,862 extractions. When limiting system tests to small dataset sizes, greater variance in system effectiveness scores might occur, leading to potentially deceptive results being produced.

4.1.2 Approach Two - KBP slot-filling task

As mentioned previously in sections 3.3.2 and 3.4.2, the OpenIE 4.0 and Stanford OpenIE systems were evaluated on the 2013 TAC KBP slot-filling task. The task consisted of a significantly large dataset and the system evaluations were carried out without any human inspection.

4.1.2.1 Advantages

Testing both OpenIE 4.0 and Stanford OpenIE on a large dataset reduces the risk of SE score variations that may occur. The effect small extractions errors have on recall and precision metrics is far greater when only few relations exist within the dataset.

4.1.2.2 Disadvantages

One of the flaws in evaluating OIE systems on the KBP slot-filling challenge is the limitations imposed on the systems. The systems were only required to find pre-defined relation types, mapped to pre-defined entities. This has the potential of not truly testing the system's full capabilities, or misleadingly testing a system that excels in one particular relation type. As an example, if extractions are limited to only noun-based relations, ReVerb would display a F_1 score of 0, a result that is not a true reflection of ReVerb's capabilities.

Another potential flaw of the KBP slot-filling task was evaluating extractions without human input. The large numbers of extractions produced by the systems

resulted in humans being unable to inspect all extractions. Consequentially, any bug or other code related issues may go unnoticed. This could have potentially harmful consequences, further contributing to inaccurate results.

4.1.3 Approach Three – Manual annotation

The final approach, outlined in section 3.5, was implemented by the team from the University of Alberta in [2]. This involved manual annotation of datasets to find all relations that exist. This annotated dataset was then used as a reference to determine the correctness of the system extractions.

4.1.3.1 Advantages

Both the advantage and disadvantage of this approach is the reliance on human input to annotate a dataset. The accuracy of this annotation process is reliant on the linguistic capabilities of the annotator. If the annotator is skilled in this area of analysis, this would improve the accuracy of system results.

4.1.3.2 Disadvantages

As mentioned previously, the linguistic capabilities of the annotator determines both the number of relations found within the dataset, and the correctness of these relations found. If the annotator lacks skill in this area of analysis, this could affect system evaluations going forward.

Additionally, the reliance on manual annotation limits the dataset size that can be used for the OIE system evaluations. Once again systems are exposed to increased variance in SE scores.

4.1.4 SE evaluation approach chosen

Ultimately approach three was implemented to evaluate the SE scores of the three OIE systems investigated in this report. This decision was brought about after the following considerations:

Consideration 1

Approach one relied on too many early dated OIE systems to find all relations within a dataset. These systems were often unable to find many relations that existed within the dataset. This could be attributed to these systems lacking the ability to extract noun-mediated relations, a result of the shallow parsing of text these systems employed.

Consideration 2

The KBP slot-filling task required extensive analysis of the datasets used to test the OIE systems on. This analysis was needed to create the pre-defined entities and relations as shown in figure 6. Our team was not able to carry out this significantly time-consuming annotation process. Additionally, the evaluation code provided by the Textual Analysis Conference (TAC) was not well documented. If evaluation problems occurred, they would be difficult to diagnose and fix. The evaluation process could become extremely complicated and be prone to significant errors.

Consideration 3

A brief inspection of the datasets annotated in [2] showed promising results. The annotations appeared to cover both verb and noun mediated relations. Additionally, using a smaller sized dataset allowed for manual inspection if errors any errors were apparent.

4.2 System Effectiveness Dataset Selection

After choosing approach three, discussed in 4.1.3, the datasets from [2] were then analysed to determine their suitability as test datasets. The three labelled datasets provided in [2] are:

1. WEB-500
2. NYT-500
3. PENN-100

4.2.1 WEB-500

The WEB-500 dataset consists of 500 sentences returned via search engine queries with known relation instances [3]. This dataset contains many incoherent sentences with improper sentence structure. Additionally this dataset contained many repeated sentences, or similar relation types, as shown in figure 10.

```
1- [[[NONE Google]]] ---> has {{{acquired}}} the video sharing website <---
[[[NONE YouTube]]]
2- [[[NONE Adobe Systems]]] ---> plans to {{{acquire}}} <--- [[[NONE
Macromedia]]]
3- [[[NONE Adobe]]] {{{acquired}}} [[[NONE Macromedia]]]
```

Figure 10 – Extracts from WEB-500 dataset [2] where [[[ ]]] represents entities annotated, {{{ }}} represents relations annotated and <--- shows allowed tokens

As a result of the sentence repetitions contained within, this dataset was not chosen for testing of the three OIE systems.

4.2.2 NYT-500

The NYT-500 dataset consists of 500 randomly extracted sentences from the New York Times Annotated Corpus [4]. This dataset is comprised of many complex sentences, often containing numerous relations. Upon further inspection, the manually labelled extractions provided in [2] presented many relations that neither we, nor the three OIE systems evaluated in this report, were able to produce. One such example is shown in figure 11.

```
The area is 65 miles east of [[[ORG Prudhoe Bay]]] , [[[LOC North
America]]] ---> 's largest {{{oilfield}}} <---
```

Figure 11 – The relation extraction tuple (Prudhoe Bay, oilfield, North America) was deemed incorrect, both from manual inspection and OIE system extractions.

As a result of these annotation errors, the NYT-500 dataset was not chosen for further system tests.

4.2.3 PENN-100

The PENN-100 dataset consists of sentences provided by Penn Treebank, a project established by a team from the University of Pennsylvania. This dataset was annotated and provided by the authors of [7]. Manual inspection showed coherent sentences, with limited repetitions. As a result, this dataset was chosen for the OIE system tests.

4.3 System Speed Dataset Selection

In addition to evaluating system effectiveness, system speeds also needed to be calculated. To evaluate system speeds, random sentences were extracted from a larger dataset, the psychology journal [22]. This dataset was comprised of a number of abstracts from various scientific papers within [22]. Various dataset sizes were created, ranging from 50 sentences (8KB) to 700 sentences (115KB).

4.4 PENN-100 Formatting

The annotated PENN-100 dataset, as shown previously in figures 10 and 11, contained various symbols mixed in among the original sentences of the PENN-100 dataset. A parsing program was created using regex patterns to replace these symbols with whitespaces, in an effort to obtain the original sentences within the PENN-100 dataset.

The original dataset was then used as an input to the three OIE systems, resulting a set of system extractions for each OIE system. The formatting process is shown in figure 12.

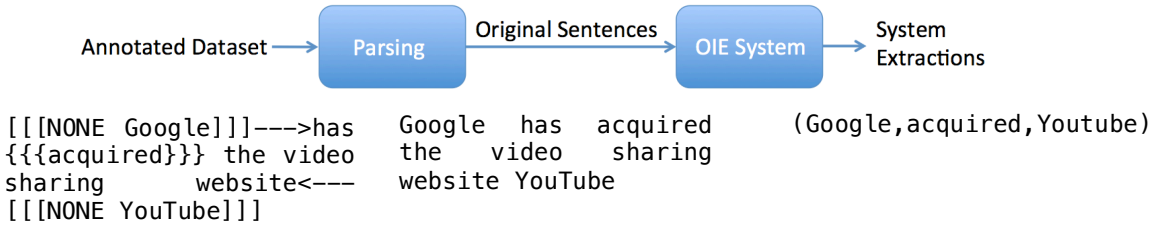

Figure 12 – The dataset formatting process used

For each OIE system, the authors released command line instructions describing how to run the systems. After initial testing, it was found that the Stanford OpenIE system was only able to operate on the larger datasets, if a minimum of 5GB of heap memory was allocated for the program. As a result all programs were allocated an additional 5GB of heap memory, in order to test system speeds in an unbiased manner.

The command line instructions for operating each OIE system, as well as the parsing program described in figure 12, is available on our github page [19].

4.5 Extraction Evaluation Process

The relation evaluation program published by the team from the University of Alberta in [2], lacked any meaningful documentation. Additionally, the format required for system extractions when using this program was different to the format of extractions produced by the three OIE systems tested. To reduce the possibility of any undetectable errors, as well as a general lack of confidence in the program presented in [2], a novel extraction evaluation program was created.

The main premise behind the novel extraction evaluation program proposed in this report, involved creating "Sentence" objects for both the annotated dataset, and the extractions produced by the OIE systems. We then compared these Sentence objects to determine the correctness of the system extractions, using the annotated sentences as a reference point.

4.5.1 Sentence Object Creation

The first step in the extraction evaluation program we designed entailed creating what we labelled "Sentence" objects. These were java-based objects with various attributes including:

SentenceID – this attribute referenced which sentence number in the dataset this extraction was taken from.

Relation – this attribute contained the relation either within the annotated dataset, or the system extractions.

Entities – this attribute contained the entities involved with the relation.

Check flag – a Boolean flag was used to indicate partial relation or entity matches. These checks are discussed in section 4.5.2.

Correct flag – a Boolean flag to indicate if the sentences compared are indeed a correct match.

Figure 13 shows the creation of two different sets of sentence objects, one for the Annotated Dataset and the other for the System Extractions.

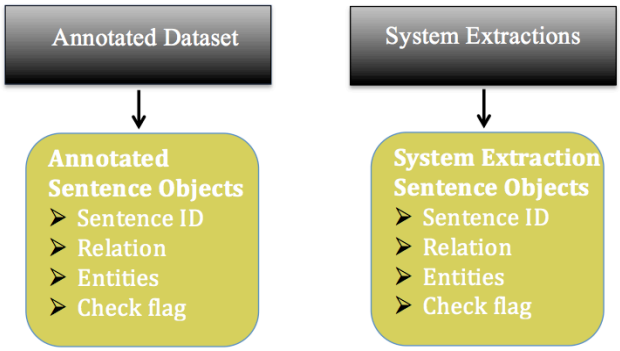

Figure 13 – Sentence object creation

Figure 14 shows a portion of the annotated sentences created from analysing the PENN-100 dataset.

Sentence No.	Relation	Entities	Correct	Checkflag
2	farmer	Ralph Holzfaster, Ogallala,	false	false
10	chairman	George L. Ball, Prudential Insurance Co.,	false	false
12	business	L.J. Hooker International, Merrill Lynch Commercial Real Estate,	false	false
17	Chairman	Boeing, Frank Shrontz,	false	false
19	unit	Heitman Advisory Corp., Heitman Financial Corp.,	false	false
21	suppliers	EC, U.S.,	false	false
23	maker	Andrew Jergens Co., Cincinnati,	false	false
25	operates	Sulka, U.S.,	false	false
28	offer	Salomon, Hongkong & Shanghai Banking Corp,	false	false
29	successor	Northview Corp., Vagabonds Hotels,	false	false
30	analyst	Daniel Basse, AgResource Co.,	false	false
32	joins	Robert Louis-Dreyfus, Saatchi,	false	false
33	agency	Omnicom Group, BBDO,	false	false
34	owns	Samsung, Korea First Advertising Co.,	false	false
36	analyst	Alan Kassan, Shearson Lehman Hutton,	false	false
37	President	UAW, Stephen P. Yokich,	false	false
39	partner	Viren Mehta, Mehta & Isaly,	false	false
44	president	Kenneth Olsen, Digital Equipment Corp.,	false	false
45	president	Edward A. Friedman, Helmsley Spear Inc,	false	false
46	trading	Columbia Laboratories Inc., COB,	false	false
49	acquire	Meridian Bancorp Inc., Hill Financial Savings Association,	false	false

Figure 14 – Visualisation of Sentence objects created

4.5.2 Sentence Comparisons

Once both annotated dataset sentences and system extractions sentences were created, the sentences were put through a flowchart process shown in figure 15.

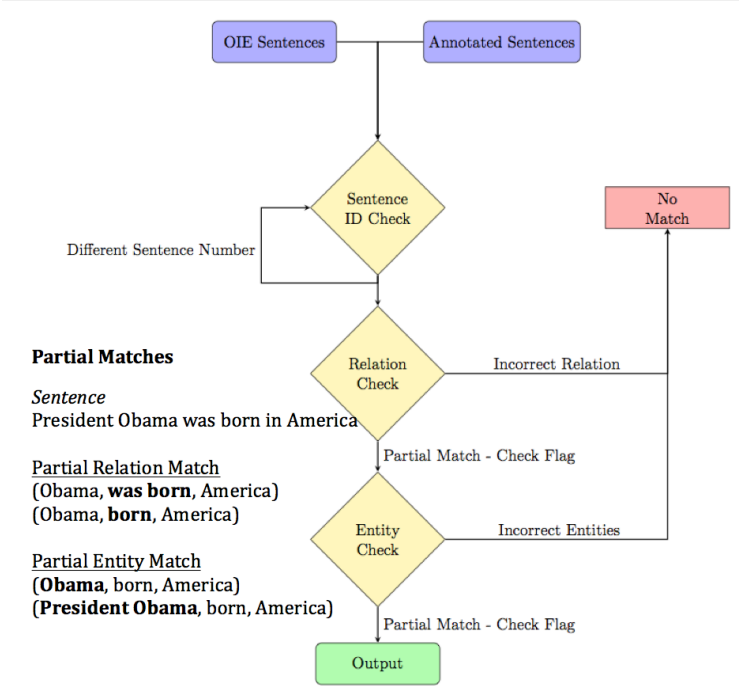

Figure 15 – Relation evaluation flowchart

The steps involved in the sentence comparison programs were as follows:

1. **Sentence ID check** – The two sentences being analysed were required to have the same Sentence ID, ensuring the proper extractions and annotations were compared to each other. This removed the possibility of false-positive matches, an occurrence that could arise if sentences were repeated in the dataset, as seen in the WEB-500 dataset.

2. **Relation check** – The relations within the annotated sentences were compared to the system extraction sentences. If one relation was a substring of the other, but not an exact string match, a partial match was deemed to have occurred. An example of such a partial match relation match is shown in figure 15. If a partial match occurs, the Check Flag attribute of the system extraction sentence is set true, indicating further analysis must take place. If no match occurs, a new sentence from either the annotated sentences or extraction sentences is chosen for testing. If an exact match occurs, the sentences proceed to the next testing stage with no Check Flag set.

3. **Entity check** – The final test the sentence objects underwent was an entity check. This required checking the entities of the sentence objects to

one another to see whether a partial or exact match occurred. This comparison was similar to the relation check described previously. If one entity was a substring of another, but not an exact string match, the Check Flag attribute was set true. An example of a partial entity match is shown in figure 15. If an incorrect match occurred, a new sentence from either the annotated sentences or system sentences was chosen for analysis.

At the end of this comparison process, all system extraction sentences were noted as either:

Incorrect – Neither Check Flag nor Correct Flag were set True.

Partially Correct – The sentence requires manual inspection to determine whether or not the extraction is correct. This is up to the discretion of the human examiner.

Correct – The sentence analysed is entirely correct.

The total number of relations within the dataset was provided by the authors in [2]. The number of correct relations extracted by the systems was the sum of the number of correct sentences, and the number of partially correct sentences deemed correct after further analysis. The total number of relations, and the number of correct system extractions were used in calculating precision and recall scores.

The Sentence Comparisons system was designed to operate on an automated principle. It was the intention that this system could be used in future OIE system evaluations. To ensure the accuracy of this system, a full manual inspection of the sentences took place afterwards to verify the results produced.

All components of the System Extraction Evaluation system are available on our github page [19].

4.6 System Speed evaluation

For the system speed evaluation process, a java program was created to extract differing numbers of sentences from one dataset source. The result was 10 newly created datasets ranging between 50 to 700 sentences. Each system was run twice on each of the newly created datasets, using the linux time command. The linux time command produces three different metrics as shown in figure 15.

```
real 1m55.506s
user 1m50.749s
 sys  0m4.274s
```

Figure 16 – Sample output from linux time command

The various runtimes; **real**, **user** and **sys** all refer to different timing metrics.

real – displays the elapsed time in seconds since the application process was started [20].

user – displays the total time in CPU-seconds that the process spent in user mode [20]. User mode refers to when the operating system is directly running the application process.

sys – displays the total time in CPU-seconds that the process spent in kernel mode [20]. Kernel mode refers to when the operating system is managing other resources, such as memory for the application process.

The timing of interest in this report is the sum of **user** and **sys**, that is the total time in CPU-seconds that the application required to run. Each OIE system was run twice on each of the 10 datasets created. The average of both runtimes was used as the final runtime.

5. Results

The results for this report have been split into two sections. The first section shows the system effectiveness results calculated for each OIE system. The second section shows the system speed results recorded for each OIE system.

5.1 System Effectiveness

The System Effectiveness (SE) scores for the three OIE systems are shown below in figure 17.

PENN-100			
OIE System	Precision	Recall	F-1 Score
ReVerb	0.777	0.137	0.233
OpenIE 4.2	0.884	0.451	0.618
Stanford OpenIE	0.71	0.529	0.606

Figure 17 - System effectiveness scores on the PENN-100 dataset

51 relations existed within the PENN-100 dataset according to the annotation provided in [2].

ReVerb - extracted 9 relations in total. 7 of these extractions were correct. ReVerb displayed the worst system effectiveness, with the lowest F_1 score of 0.233.

OpenIE 4.2 - extracted 26 relations in total. 23 of these extractions were correct. OpenIE 4.2 displayed the best system effectiveness, achieving the highest F_1 score. Additionally, OpenIE 4.2 had the best precision, indicating it produces the highest percentage of correct extractions.

Stanford OpenIE - extracted 38 relations in total. 27 of these extractions were correct. Stanford OpenIE was the second best performing OIE system in terms of F_1 score. The Stanford OpenIE system did however produce the largest amount of correct extractions out of the three systems, with 27 correct extractions. This results in the Stanford OpenIE having the largest recall score of 0.529.

These results were validated in part through comparison to the results produced in [2], shown in figure 18. The team from the University of Alberta achieved the same results when testing ReVerb on the PENN-100 dataset.

Method	PENN-100			
	Time	P	R	F
ReVerb	**0.02**	0.78	0.14	0.23
SONEX	0.04	**0.92**	0.43	0.59
OLLIE	0.14	0.81	0.43	0.56
EXEMPLAR[M]	0.16	0.83	0.49	**0.62**
EXEMPLAR[S]	0.62	0.79	**0.51**	**0.62**
PATTY	0.66	0.46	0.24	0.31
SwiRL	2.17	0.89	0.16	0.27
Lund	5.21	0.86	0.35	0.50
TreeKernel	0.85	0.85	0.33	0.48

Figure 18 - System effectiveness scores documented in [2]

5.2 System Speed

Shown in figure 19 are the average runtimes for each OIE system on the 10 different datasets mentioned previously in section 4.3. The runtimes are given in CPU-seconds, the amount of time in seconds that the CPU spent on the application process, or necessary Operating System (OS) tasks. This metric allows for effective comparisons to be made, instead of relying on real world times, which may include the OS scheduling non-related tasks.

Runtime - CPU Seconds			
Dataset Size(Sentences)	ReVerb	OpenIE 4.2	Stanford OIE
50	0m10.001s	1m0.674s	0m52.631s
100	0m17.243s	1m2.375s	1m24.747s
150	0m18.471s	1m10.767s	2m1.941s
200	0m20.65s	1m18.003s	2m8.779s
250	0m21.025s	1m18.45s	2m34.2885s
300	0m21.828s	1m23.615s	3m15.727s
350	0m23.127s	1m30.851s	3m43.711s
500	0m25.557s	1m44.134s	5m26.911s
600	0m24.86s	1m55.023s	7m29.267s
700	0m27.284s	2m13.773s	8m31.674s

Figure 19 - System Runtimes

ReVerb displayed the fastest runtimes out of the three OIE systems. ReVerb also showed the shortest initialisation time, indicated by the runtime when processing the smallest dataset size of only 50 sentences.

OpenIE 4.2 displayed the longest initialisation time with a runtime of 1 minute and 0.674 seconds on the 50-sentence dataset. OpenIE 4.2 showed relatively small runtime increases on the larger datasets when compared to Stanford OpenIE. On the largest dataset, OpenIE 4.2 ran 3.82 times faster than the Stanford OpenIE system.

Stanford OpenIE displayed the worst system speed out of the three OIE systems. Despite not having the longest initialisation time, the Stanford OpenIE system ran significantly slower than the other systems as the dataset size was

increased. On the largest dataset, Stanford OpenIE ran almost 19 times slower than ReVerb and 4 times slower than OpenIE.

Shown in figure 20 is a visualisation of the system runtimes shown previously in figure 19. As can be seen the Stanford OpenIE performs significantly slower than the other OIE systems as the dataset size is increased.

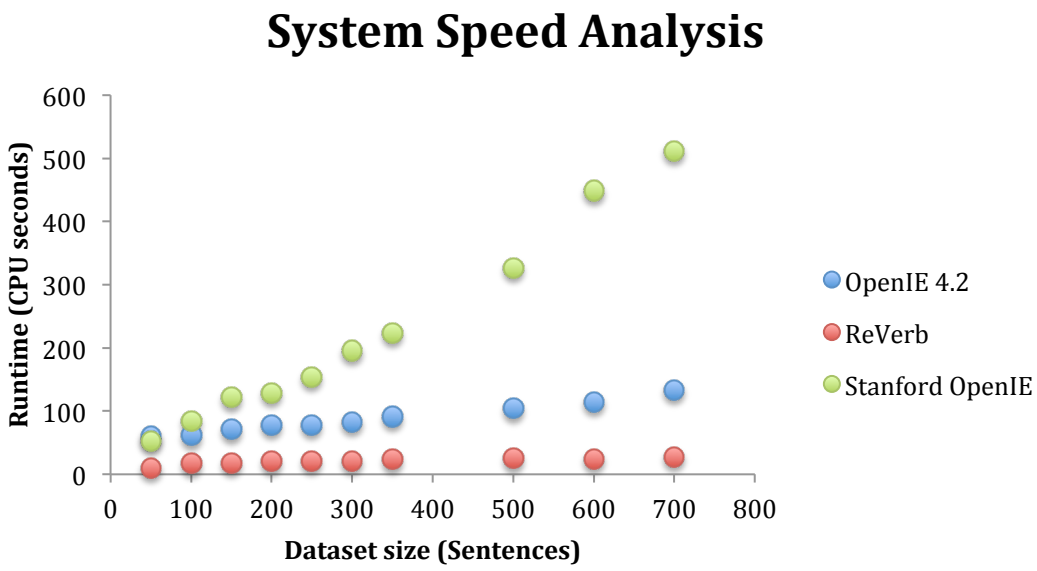

Figure 20 - Visualisation of System Runtimes

6. Discussion

The following section explains the significance of the results published in section 5. Additionally, this section contains other insights realised in this report.

6.1 Effectiveness & Speed Trade-off

As mentioned previously in [2][15][16], there often exists a trade off between system effectiveness and system speed. In this report, ReVerb achieved the lowest F_1 score, indicative of its reliance on shallow parsing. This shallow parsing was done through Part of Speech (POS) tagging and Named Entity Recognition (NER). OpenIE 4.2 and Stanford OpenIE however, achieved significantly greater F_1 scores, a result of implementing a dependency parser. A dependency parser provides a deeper parsing of input sentences, providing more information than either POS tagging or NER analysis. The consequence of an OIE system being reliant on a dependency parser, is a significant increase in system runtime. This was evident in both OpenIE 4.2 and Stanford OpenIE running 4.9 and 18.8 times slower than ReVerb respectively, when tested on a dataset size of 700 sentences. It is apparent from figure 20, that runtimes of OpenIE 4.2 and Stanford OpenIE are inflated further when the systems are exposed to larger datasets.

The largest contributing factor to the F_1 score discrepancies between the three OIE systems was the recall metric. All systems displayed similar precision values, varying between 0.71 and 0.884, however a wider range of 0.137 and 0.529 in recall values was seen. This variance in recall can be attributed to ReVerb's limitation in extracting only verb-mediated relations. The PENN-100 dataset was shown to contain many noun-mediated relations as shown in figure 21.

```
[[[NONE South Korea]]] ---> 's {{{President}}} <--- [[[NONE Roh]]] traveled
to the U.S. for a five-day visit that is expected to focus on ties between
Washington and Seoul
```

Figure 21 – A noun-mediated relation in the PENN-100 dataset

Employing a dependency parser in an OIE system, allows for noun-mediated relations to be extracted, the result of implementing deeper textual analysis. Therefore, ReVerb was only able to extract 9 relations in total, compared to the 26 and 38 relations extracted by OpenIE 4.2 and Stanford OpenIE respectively.

These results are significant for future OIE system designs. OIE systems need to be designed according to their functional requirements, whether they are used in search engines, question answering systems, textual summation systems or any other purpose. OIE system designers need to choose between system effectiveness and system speed.

6.2 Dependency Parser Efficiency

Interestingly, despite OpenIE 4.2 and Stanford OpenIE implementing a dependency parser, OpenIE 4.2 was able to run substantially faster than Stanford OpenIE. Whilst OpenIE 4.2 required a larger initialisation time, it displayed far

better system speeds on larger datasets. Further investigation regarding these findings are recommended as future work in section 7.

6.3 Need for Standardisation

Evident throughout this report was the need for standardisation of both datasets, and extraction evaluation methodologies, within the OIE research field. As discussed in section 4, there exist three different extraction evaluation methods. Additionally, a number of different datasets have been used in previous OIE system evaluations, each annotated by different authors. These different methods and datasets make OIE system comparisons a difficult task.

Different extraction evaluation methodologies introduce differing opinions between researchers as to what constitute absolute relations. What one researcher may deem to be a correct extraction, another may disagree. This difference in opinion is embedded in the different extraction evaluation methodologies created, making reputable system comparison difficult to be carried out.

Testing datasets that are annotated by different authors makes it difficult to see how different systems perform on particular types of data. What some authors may consider a relation, others may disagree. This disagreement may show a bias in system performance, where authors are able to tailor dataset annotation to match their system's abilities. When these systems are then reported on, their effectiveness can be misleading for future users.

The lack of documented code is also a problem within the OIE research field. Neither the testing code provided in [2] nor [5] is easily understandable or reproducible. This make errors difficult to diagnose and repair, making these testing protocols impractical to employ. There is also no evidence provided for the application user to trust the program they are running, if very little documentation is provided.

This need for standardisation, both in terms of dataset annotation, extraction evaluation methodology and testing code is recommended as future work in section 7.

7. Conclusion & Future Work

In this report we evaluated three state of the art OIE systems and documented our findings. In section 4 we outline our methodology, beginning with our decision regarding the system extraction evaluation we implemented. Once an evaluation method was chosen, we investigated which dataset would be used to test the three OIE systems. Ultimately the PENN-100 dataset provided by the Penn-Treebank project was chosen. We then presented a novel approach towards evaluating extractions produced by OIE systems. Our code is available on our github page [19]. It was found that ReVerb was the best performing system with regards to system speed, whilst OpenIE 4.2 displayed the greatest system effectiveness. These results were attributed to ReVerb's shallow parsing technique, whist OpenIE 4.2 and Stanford OpenIE relied on deeper textual analysis. This deeper textual analysis was carried out by implementing a dependency parser for pre-processing input sentences.

In section 6.4 we highlighted the need for standardisation of both annotated datasets, and extraction evaluation methods within the OIE research community. We recommend future research go toward creating several standard annotated datasets. This research must focus on defining what constitutes a relation, as well as defining what constitutes a match between a system extraction, and an annotated relation. Part of this future work is the creation of a standard system extraction evaluation method, with fully detailed and transparent code. We believe it is also important to implement a standardised format for system extractions as well as annotated datasets. This standardised format would make future OIE system comparisons easier for researchers, as no data formatting would be required for OIE system tests.

Additionally, there still exists room for further testing of the three OIE systems investigated in this report. Additional dataset testing would either confirm or challenge the results presented in this report. We propose that future datasets tested, vary both in terms of language style and size. We are also interested in investigating datasets that contain n-ary relations. Out of the three systems tested in this report, only OpenIE 4.2 is able to extract n-ary relations. Investigating how much of an effect this has on system effectiveness scores would be an interesting endeavour. It would allow future OIE system designers to consider the value in creating a system that can extract n-ary relations. If the ability to extract n-ary relations significantly slows the system down, whilst only slightly improving it's F_1 score, then it may not be worth implementing.

Finally, we recommended additional research into the different components used in the three OIE systems. Future OIE system design will require using the latest state of the art NLP programs, such as the most up to date POS taggers, NER programs and dependency parsers. One area of interest is investigating what effect upgrading the current NLP programs, implemented by three OIE systems, has on system effectiveness and runtime. It is also recommended to further investigate OpenIE 4.2, specifically its reliance on the dependency parser it uses. Despite both OpenIE 4.2 and the Stanford OpenIE systems implementing a dependency parser, OpenIE 4.2 is able to produce much faster system

runtimes. Discovering how this is achieved would be extremely valuable for future OIE system designs. Alternatively, instead of researching how to improve system speeds of second generation OIE systems, it may be worth researching how to improve system effectiveness in first generation OIE systems. Future work regarding the ability for shallow parsing systems such as ReVerb, to be able to extract noun-mediated relations, could be crucial when designing OIE systems. As OIE systems are designed to operate on large coropora of data, system speed is an extremely important system characteristic. Increasing system speed is going to be even more prevalent, if OIE systems are to be used in real-time search engines.

8. Acknowledgements

I would like to thank Prof. Roberto Togneri for his help and guidance on this final year project.

9. References

[1]D. Vo and E. Bagheri, "Open Information Extraction", *ENCYCLOPEDIA WITH SEMANTIC COMPUTING*, vol. 1, no. 1, 2016.

[2]M. Filipe, J. Schmidek and D. Barbosa, "Effectiveness and efficiency of open relation extraction", *New York Times*, no. 500, 2013.

[3]M. Banko, M. Cafarella, S. Soderland, M. Broadhead and O. Etzioni, "Open Information Extraction from the Web", *IJCAI*, vol. 7, pp. 2670-2676, 2007.

[4]Sandhaus, Evan. The New York Times Annotated Corpus LDC2008T19. DVD. Philadelphia: Linguistic Data Consortium, 2008.

[5]G. Angeli, M. Premkumar and C. Manning, "Leveraging Linguistic Structure For Open Domain Information Extraction", *Proceedings of the Association of Computational Linguistics (ACL)*, 2015.

[6]M. Surdeanu, Surdeanu, Mihai. "Overview of the TAC2013 knowledge base population evaluation: English slot filling and temporal slot filling", *Proceedings of the Sixth Text Analysis Conference (TAC 2013)*, 2013.

[7]F. Mesquita, J. Schmidek and D. Barbosa, "Effectiveness and efficiency of open relation extraction", *New York Times*, vol. 500, p. 150, 2013.

[8]A. Fader, L. Zettlemoyer and O. Etzioni, "Open question answering over curated and extracted knowledge bases", *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1156-1165, 2014.

[9]O. Etzioni, "Search needs a shake-up", *Nature*, vol. 476, no. 7358, pp. 25-26, 2011.

[10]S. Sarawagi, "Information Extraction", *Foundations and Trends in Databases*, vol. 1, no. 3, pp. 261–377, 2007.

[11]F. Wu and D. Weld. “Open information extraction using Wikipedia”. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 118–127, 2010.

[12]O. Etzioni, A. Fader, J. Christensen, S. Soderland, and Mausam. “Open information extraction: The second generation”, *Proceedings of the Conference on Artificial Intelligence*, pp. 3–10, 2011.

[13]A. Akbik and J. Broß. “Wanderlust: Extracting Semantic Relations from Natural Language Text Using Dependency Grammar Patterns”, *1st Workshop on Semantic Search at 18th*. WWWW Conference, 2009.

[14]A. Akbik and A. Lo¨ser. “Kraken: N-ary facts in open information extraction”, *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pp. 52–56, 2012.

[15]M. Schmitz, R. Bart, S. Soderland and O. Etzioni, "Open language learning for information extraction", *EMNLP-CoNLL '12 Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 523-534, 2012.

[16]L. Del Corro and R. Gemulla, "ClausIE: Clause-Based Open Information Extraction", *Proceedings of the 22nd international conference on World Wide Web - WWW '13*, 2013.

[17]A. Fader, M. Schmitz, M. Banko, M. Cafarella, O. Etzioni, R. Bart and S. Soderlan, "Open IE by knowitall", *Knowitall.github.io*. [Online]. Available: https://knowitall.github.io/openie/. [Accessed: 21- October- 2016].

[18]A. Fader, S. Soderland and O. Etzioni, "Identifying relations for open information extraction", *EMNLP '11 Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2011.

[19]J. Weiner, "Jake-Weiner/A-Comprehensive-Evaluation-of-Three-Well-Recognised-OIE-Systems", *GitHub*, 2016. [Online]. Available: https://github.com/Jake-Weiner/A-Comprehensive-Evaluation-of-Three-Well-Recognised-OIE-Systems.git. [Accessed: 30- Oct- 2016].

[20]A. Brouwer, *time.1*. http://man7.org/linux/man-pages/man1/time.1.html: Linux, 2000.

[21]"The Apache Software Foundation", *Apache.org*, 2016. [Online]. Available: http://www.apache.org/. [Accessed: 29- Oct- 2016].

[22]"Judgment and Decision Making, Journal Home Page", *Journal.sjdm.org*, 2016. [Online]. Available: http://journal.sjdm.org/. [Accessed: 30- Oct- 2016].

[23]Silveira, N., Dozat, T., De Marneffe, M. C., Bowman, S. R., Connor, M., Bauer, J., & Manning, C. “DA Gold Standard Dependency Corpus for English”, *LREC* ,pp. 2897-2904, 2014.