

Extracting data governance information from Slack chat channels

By
Simon Quigley

Supervisor: Dr. Rob Brennan

Assistant supervisor: Dr. Alfredo Maldonado

Dissertation

Presented to University of Dublin, Trinity College
in fulfillment of the requirements for the Degree of

Master of Science in Computer Science

University of Dublin, Trinity College

May 2018

Declaration

I, Simon Quigley, declare that the following dissertation, except where otherwise stated, is entirely my own work; that it has not previously been submitted as an exercise for a degree, either in Trinity College Dublin, or in any other University; and that the library may lend or copy it or any part thereof on request.

Signature: _____

Date: _____

Abstract

Extracting data governance information from Slack chat channels

By Simon Quigley

Data governance is increasingly important in organisations, and formal systems of data governance have become widespread. Large amounts of communication within these organisations, including data governance information, is carried out over chat channels such as Slack, and this data governance information may not be captured in the organisations data governance system. This dissertation set out to evaluate the effectiveness of a named entity recogniser (NER) in extracting this data governance information from such chat channels. To do this, a dataset of chat messages annotated with data governance related entities was created and a NER optimised and evaluated using this data. Results of this evaluation were promising given the small size of the dataset, but found a high disparity in the named entity recogniser's performance between the annotated entity types. An analysis of factors that affected the NER's performance on each entity type was carried out, finding the type-token ratio and the number of occurrences of an entity type to be a good indicator of the NER's performance on that entity type. The mean and standard deviation of the word length of an entity type were not found to be correlated with the NER's performance. The overall performance of the NER demonstrated that named entity recognition can effectively extract data governance information from chat messages, and future iterations trained on a larger dataset could potentially achieve very high accuracies.

Acknowledgements

I would like to give my sincerest thanks to my supervisor Rob Brennan and assistant supervisor Alfredo Maldonado, for all the help and advice they gave throughout this dissertation.

I would also like to thank my family for all their support throughout my time spent in college.

Contents

Abstract.....	iii
Acknowledgements.....	iv
List of Figures.....	vii
1. Introduction.....	1
1.1. Motivation.....	1
1.2. Research Question.....	3
1.3. Contribution.....	3
1.4. Technical Approach.....	4
1.5. Document Summary.....	5
2. Background.....	7
2.1. Data Governance.....	7
2.2. Evaluation Methods.....	8
2.2.1. Performance Metrics.....	8
2.2.2. Resampling Methods.....	9
2.2.3. Type-Token Ratio.....	10
2.3. NER Algorithms.....	10
2.4. Tools.....	11
2.4.1. Stanford Named Entity Recogniser.....	11
2.4.2. Brat rapid annotation tool.....	12
2.4.3. Scripts.....	12
3. State of the Art.....	14
3.1. Natural Language Processing in Data Governance.....	14
3.2. Related NER Applications.....	16
3.2.1. Domain Specific NER.....	16
3.2.2. Short and Informal Message NER.....	17
3.3. Collibra Data Governance Operating Model.....	19
3.3.1. Structural Concepts.....	20
4. System Design.....	22
4.1. Dataset.....	22
4.1.1. Data Source.....	22
4.1.2. Annotation Scheme.....	22
4.1.3. Data Preparation.....	24

4.1.4. Annotation Process.....	25
4.1.5. Dataset Properties	26
4.2. Experimentation and Evaluation Software	28
4.2.1. Initial NER System Experimentation	28
4.2.2. Automation Scripts.....	28
4.2.3. Metrics	30
4.2.4. NER Experimentation	31
4.2.5. Plotting.....	32
5. Evaluation.....	35
5.1. Introduction	35
5.2. Hypothesis	35
5.3. Methodology	37
5.3.1. Cross Validation Approach.....	38
5.4. Results	40
5.5. Analysis.....	42
5.5.1. Impact of Number of Instances Annotated on NER Accuracy.....	43
5.5.2. Impact of Type-Token Ratio on NER Accuracy	45
5.5.3. Impact of Mean Entity Length on NER Accuracy	48
5.5.4. Impact of Standard Deviation of Entity Length on NER Accuracy	51
5.5.5. Summary	53
5.6. Conclusion	54
6. Conclusions	56
6.2. Future Work	57
7. Appendices.....	58
7.1. Data Annotation Notes	58
7.1.1. General ideas	58
7.1.2. Example Annotation Process (Think out loud)	59
8. Bibliography	62

List of Figures

<i>Figure 2.1 Entity annotation in brat.....</i>	<i>12</i>
<i>Figure 3.1 Collibra Data Governance Operating Models Main Asset Types.....</i>	<i>20</i>
<i>Figure 3.2 Collibra Data Governance Operating Models asset types and relations</i>	<i>21</i>
<i>Figure 4.1 Word count & Unique word count of each entity type.....</i>	<i>27</i>
<i>Figure 4.2 Mean and Standard Deviation of the length of each entity type.....</i>	<i>27</i>
<i>Figure 4.3 Python function to train and test Stanford NER CLI.....</i>	<i>29</i>
<i>Figure 4.4 Evaluation script configuration file.....</i>	<i>30</i>
<i>Figure 4.5 Metrics script output excerpt.....</i>	<i>31</i>
<i>Figure 4.6 The best performing Stanford NER configuration.....</i>	<i>32</i>
<i>Figure 4.7 Scatter plot generation code.....</i>	<i>34</i>
<i>Figure 5.1 Experimental Protocol.....</i>	<i>37</i>
<i>Figure 5.2 Results from 4-fold Validation for each entity type.....</i>	<i>41</i>
<i>Figure 5.3 Correlation between number of annotated instances and performance of an entity type.....</i>	<i>43</i>
<i>Figure 5.4 Correlation between type-token ratio and performance of an entity type... ..</i>	<i>45</i>
<i>Figure 5.5 Correlation between entities mean length and performance of an entity type.....</i>	<i>48</i>
<i>Figure 5.6 Correlation between standard deviation of entity lengths and performance of an entity type.....</i>	<i>51</i>

1. Introduction

This chapter gives an overview of and introduction to this dissertation. It describes the motivation for this work, the research question investigated, what contributions it made, the technical approach to the work and an overview of the remaining chapters.

1.1. Motivation

The role of Data Governance, or the management of Data Management, inside any organisation is becoming increasingly important. As the scale of information within every organisation has grown exponentially in recent years, large amounts of sensitive and valuable data and metadata are being generated and stored. The consequences of mismanagement of this data can be severe, as has been seen recently in data breaches of major organisations such as Facebook and Equifax.

Current Data Governance solutions can be difficult to utilise, limiting user access to data according to strict and inflexible rules. They employ potentially complex operating models with various assets, attributes and relations representing the data, technology, or other assets to be governed, how they should be governed, and who governs them. While the centralised, hierarchical nature of data governance is useful for controlling data access and allowing for easier data audits, this limits the degree to which data governance systems can be simplified

This complexity can lead members of an organisation to misuse the data governance solution, or avoid using it at all. As an alternative, they will often use chat channels such as Slack, which have increasingly been adopted as a primary communications channel within organizations, in order to give and request access to data or metadata. Even members with good knowledge of the data governance solution may not make adequate use of it if it is slow and cumbersome to use. Even the most intuitive and easy-to-use data governance solution will likely not be fully utilised, as issues and changes will first be discussed in a chat, and important elements of this may be lost whenever it is transferred to the data governance solution.

To avoid this loss of valuable information within the chat, it would be ideal to integrate the data governance solution with the chat with minimal interaction required by users. To achieve this, the extraction of key information from these chat messages is necessary. However this is a difficult task and a system will need to accurately extract information to ensure a relatively complete and accurate record of chat information.

Named entity recognition is an information extraction technique that has been applied very successfully to tasks involving the extraction of named entities from newspaper articles, medical papers, and many other texts. Application of this technique to chat channels in an organisation could allow key entities related to data governance to be identified and labelled.

However, most named entity recognition tasks have involved formal texts, with correct sentence structures, spelling and capitalisation. These features in a text are utilised by named entity recognisers to help in labelling entities, and without their consistent presence, NERs may struggle to accurately label entities. (Baldwin et al. 2015)

Another potential issue is the inconsistent and context dependent nature of references to the entities in this domain, particularly when referred to informally. Depending on the context, the same word or sequence of words could be referring to different types of assets, and can be difficult to infer, often even for a human labelling the data. While this is an issue in almost all texts, it appears to be particularly prevalent in the data governance domain, where distinct entities such as a database, the machine containing the database and the information stored within the database may all be referred to by the same identifier, i.e. a database of bank customers, the server hosting this database, and the actual bank customers could all be referred to by 'Bank Customers'. This problem is exacerbated by informal text, where the use of clear and proper names is less likely to occur.

However, if these problems are addressed, a named entity recogniser capable of extracting data governance information from chats could be used to better integrate enterprise chat channels with the organisations data governance solution.

1.2. Research Question

With this motivation in mind, the following research question is proposed:

To what extent can CRF-based Named Entity Recognition be used to extract data governance information from an enterprise chat channel?

In the context of this dissertation, data governance information is defined as a set of data governance assets, rules and users. Details of these entities can be found in [section 4.1.2](#).

This research question leads to the following research objectives:

Research Objective 1: Survey the state-of-the-art to select the most appropriate methods and tools to be used in this work.

Research Objective 2: Create a corpus of enterprise chat data with data governance entities annotated, upon which to train a Named Entity Recogniser.

Research Objective 3: Evaluate and optimise the labelling accuracy of a general Named Entity Recogniser trained on this data.

1.3. Contribution

In completing these research objectives, the following contributions were made:

An annotated corpus of data governance chat data with eight data governance related entity types annotated based on the Collibra Data Governance Operating Model, a meta-model for data governance. Relationships between entities are also defined in the corpus.

The annotation scheme used to create this dataset is also a contribution of this work. It defines the entity types and relations used in annotating the dataset as well as providing guidelines and examples for annotation. This will aid any future annotation efforts in this field in producing consistently annotated data.

A trained Named Entity Recogniser trained on the created annotated corpus, capable of identifying and labelling data governance information in enterprise chat messages.

An analysis of the correlation of various properties of entity types with metrics of the NERs performance on those data types. This sheds some light on the factors in a dataset influencing a NERs performance, and the investigated factors could be used to estimate the suitability of named entity recognition for identifying entity types in a dataset. This analysis could also inform data annotation schemes and practices in order to maximise NER performance for a given task.

These contributions, in particular the annotated corpus and its annotation scheme, could be of benefit to anyone looking to perform named entity recognition. The ADAPT centre at TCD in collaboration with Collibra, who assisted with this dissertation, have particular interest in the contributions of this dissertation and are likely to use and expand upon the work in dissertation in the future.

1.4. Technical Approach

In order to create the corpus of annotated data required to train a NER, a dump of chat data from a data governance team was obtained from Collibra¹, a data governance company. We then used a binary classifier previously developed by Jaimin Shah during an internship with Collibra (Shah 2017) to filter out the messages not related to data governance entities. This left a subset of chat messages suitable for manual annotation.

The entity and relation types chosen for annotation were based on Collibra's own data governance operating model², which covers key asset types, relations and governance rules such as business assets or data assets. The annotation scheme incorporated Collibra's 5 basic asset types, their data governance role type and their data user type, and added an additional entity type for data management tasks, which was not directly referenced in their operation model, but were frequently referenced in chat logs and potentially important to record. A guideline document was created to describe these entities and relations, and how they should be used.

¹ <https://www.collibra.com>

² <https://compass.collibra.com/display/COOK/Data+Governance+Operating+Model>

The chosen subset of data was then formatted for annotation, which included cleaning of some character corruption caused by encoding changes and the separation of entities such as full stops and brackets. BRAT³ (Stenetorp et al. 2012), a web annotation tool, was then used to collaboratively annotate the text in accordance with the guideline document.

This annotated dataset was then used for training and evaluating our NER. Our NER is based on Stanford NER (Finkel et al. 2005), a part of the Stanford CoreNLP package.

To fairly compare different NER configuration results, we set up a standard testing procedure. A Python script was created to comprehensively evaluate the performance of the NER as a whole, as well as a detailed breakdown of its performance for each entity type.

1.5. Document Summary

The remainder of this document is broken down into the following chapters:

Background - This chapter explores some relevant fields and technologies that are important to understand for this dissertation. An overview is given of data governance, the evaluation methods used, the range of named entity recognition algorithms available, and some of the tools used in this work. How these are relevant to the work and why they were chosen is also explored.

State of the Art - This chapter covers relevant research at the forefront of fields related to this project. Covered is the only known prior work involving natural language processing on data governance data, research investigating means of performing named entity recognition on similar domain specific and informal text, and the data governance model used in this work. How they relate to and influenced this dissertation is explained in detail.

System Design - This chapter details the implementation of this project, split into dataset and evaluation software. Dataset covers the dataset source, the annotation scheme & process, and the properties of the final annotated dataset. Evaluation

³ <http://brat.nlplab.org/index.html>

software details the tools and scripts used in training, evaluating and gathering metrics on the NER system.

Evaluation - covers the evaluation performed as part of this work. Detailed is the experiments hypothesis, correlating a NER systems performance with the complexity of an entity type, the methodology of experiment testing this hypothesis, the NER performance metrics gathered from this experiment, an analysis of these results correlating them with properties of the entity types, and the conclusions drawn from this analysis.

Conclusions - Final remarks on the dissertation, reflecting upon its implementation with respect to the initial goals set out, and suggesting potential means of expanding upon this work in the future

2. Background

This chapter will explore the technologies and tools that are most relevant to this dissertation. Section 2.1 describes data governance. Section 2.2 explores methods of evaluating the performance of a Named Entity Recogniser. Section 2.3 compares a variety of the most ubiquitous algorithms which are used for Named Entity Recognition. Finally section 2.4 details the most relevant tools that were used over the course of this dissertation and why they were chosen.

2.1. Data Governance

Data Governance can be defined as the exercise of authority, control and shared decision making over the management of data assets. Data Governance within an organisation seeks to unify the decision making of how data is managed as well as establishing who holds the rights to make these decisions, and who is held accountable for them. (Khatri & Brown 2010)

The range of decisions covered by data governance can be broken into 5 related decision domains:

Data Principles, which define the role that data plays in an organisation and how it may be useful.

Data Quality, which establishes a standard which data must adhere to in order to be used for its intended purpose, i.e. accuracy, timeliness, completeness and credibility. The metrics used to determine data quality are also established in this domain.

Metadata, which defines what information about the data collected is documented, with the goal of maximising its interpretability for its intended uses. This domain also defines how this metadata is collected and kept up to date.

Data Access, which establishes the importance of data, how access to the data should be controlled, and the means of ensuring data security and integrity.

Data Lifecycle, which determines how data is produced and retained, and when it should be retired.

Each of these domains may have a different decision maker or decision makers experienced in that domain, and who may be held accountable for any decisions made.

Many models of data governance exist, with an ISO standard for data governance existing, however there has been relatively little focus on data governance in academia. Significant advancements in the field of data governance have been made internally within organisations, which means that much of this research is not publically available. Collibra, an organisation specialising in data governance, collaborated in this work and their data governance model was used a reference in creating the annotation scheme used for annotating the chat message corpus. This data governance model is covered in detail in [section 3.3](#).

2.2. Evaluation Methods

2.2.1. Performance Metrics

The performance of NERs are generally evaluated using Precision, Recall and F-Scores, with F-1 being the most commonly used. (Hripcsak & Rothschild 2005)

Precision and recall are function of true positives in the NERs predictions and false positives in the case of precision, or false negatives in the case of recall. F-Score is a weighted harmonic mean of precision and recall, with F value of less than 1 placing greater importance on precision, and greater than 1 placing greater importance on recall. An F value of 1 places equal importance on both metrics. (Powers 2011)

For Named Entity Recognition the true positives, true negatives, false positives and false negatives are calculated for each entity type e.g. NAME, ORG, PLACE. For instance, to calculate these for entity type NAME, true positives are the count of words annotated as NAME which the NER predicted to be NAME, false positives are words predicted as NAME which were not annotated NAME, true negatives are words not annotated as NAME and were also not labelled as NAME, and false negatives are words annotated as NAME but not predicted to be NAME.

This process can then be repeated for each entity type and the results summed in order to calculate the precision and recall for the total dataset.

However one property of these performance measures is that if, for instance, the NER predicts a token to be NAME, but it was annotated as ORG, it will count as a false positive for NAME as well as a false negative for ORG.

This double counting of any misprediction of the NER, except for a misprediction of 'O' (standard annotation for 'not an entity'), which causes those mispredictions to negatively affect both precision and recall means that those mispredictions are more heavily weighted. Whether this is the best and most correct method is a matter of dispute, however it is the most commonly used method of calculating precision and recall in NER tasks. (Makhoul et al. 1999)

Metrics specifically designed for NER evaluation exist, which may give scores more representative of the NERs performance, however these are not frequently used. (Jannet et al. 2014) Due to the rarity of such metrics use, it would more appropriate to use the standard precision, recall and F1 scores for clarity and ease of comparison.

2.2.2. Resampling Methods

To get a good estimate of the NERs performance on a training set, a resampling method is used. The most common of these are Cross Validation and Bootstrap.

In k-fold cross validation, the dataset is divided into k equally sized sections. Each of the k sections is used as the test set once, with the remaining k-1 sections used as the training set. This results in k test results which are averaged to get a performance estimate of the model. Larger values of k results in a smaller test set and larger training set for each fold.

Bootstrap takes a random sample from a dataset with n data points n times, with replacement. This forms the training dataset, which may contain duplicate samples taken from the dataset. The test set consists of any data point in the dataset that were not taken as a sample for the training set. This will typically be approximately 30% of the dataset.

For small data sets, cross validation seems to be preferable due to its low variance and generally low bias, whereas bootstrap can potentially have a very high bias depending on the dataset. Bias is a general tendency for the resampling method to over or underestimate the performance of a classifier, while variance is the degree to which results can vary between runs, typically measured by the standard deviation of the resampling methods' estimates over many runs.

2.2.3. Type-Token Ratio

In the evaluation of the NER, one chosen measure of an entity type's complexity is the type-token ratio (Richards 1987). This is defined as the number of different words (types) divided by the total number of words (tokens) in a given text. This measure is most commonly used in psychology to measure the variation of vocabulary in a person's speech or writing.

Type-token ratio is not commonly used in the field of named entity recognition, but NERs typically have much higher accuracy on words they have previously encountered. Type-token ratio was hence chosen as it is a good numeric representation of the number of repeated words in a given text.

2.3. NER Algorithms

A large number of algorithms exist that can be used for named entity recognition, most of which have many other applications in machine learning.

The oldest and most basic method is a **rule-based classifier**, where the rules for identifying entities are written by hand. This is very labour intensive, and cannot generalise at all to new data, however it has the benefit of not requiring any training data.

Some of the earlier machine learning NERs made use of **Hidden Markov Models** (Smith 2011, p.74) for classification. These were fast and achieved reasonably good accuracies, however they cannot take into account neighbouring entities for classification, and they model a joint distribution which is not considered necessary or optimal for classification tasks. They have therefore been replaced by more recent and computationally intensive algorithms for most tasks.

The most commonly used algorithm in modern NERs is **CRF (Conditional Random Fields)** (Smith 2011, p.89). CRF is closely related to Hidden Markov Models, however there are several key differences that make it superior for NER tasks. CRFs model a conditional distribution, which is more suited to classification tasks than a joint distribution. They can also take into account previous and future labels when labelling an entity, and are more capable of handling imbalanced data. CRF based NERs are however significantly slower to train than HMM based NERs.

The current state of the art algorithm in named entity recognition tasks are **Neural Networks**, with the best NERs using Recurrent Neural Network based feature extractors outperforming the best CRF based NERs utilising traditional feature extractors, though generally by relatively small margins (Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio n.d.).

While NERs using neural network based feature extractors have the potential to be the most accurate currently available, they require a large data corpus to achieve their performance, and the massive number of hyperparameters that can be tuned make it very difficult to find an optimal network for a given task.

2.4. Tools

2.4.1. Stanford Named Entity Recogniser

The Named Entity Recogniser used in this project was based on the Stanford Named Entity Recogniser. (Finkel et al. 2005) This is a widely used open source implementation of a Named Entity Recogniser, and it has been shown to have better performance than other well-known publicly available NER tools such as Illinois Named Entity Tagger and OpenCalais Named Entity Recogniser in discursive texts such as biographies. (Atdağ & Labatut 2013)

Stanford NER is written in Java and is highly configurable through configuration files, with further modification possible via extension of the freely available source code. The NER utilises conditional random fields as its modelling method. It includes many built in feature extractors to enhance performance, such as character level n-grams which allow identification of common prefixes and suffixes of entity types. (Klein et al. 2003)

As a part of the Stanford CoreNLP set of language processing tools (Manning et al. 2014), Stanford NER is designed to integrate smoothly with other tools in the set such as a part-of-speech tagger or relationship extractor. Relevant for this project is the Stanford tokeniser which formats text and splits it into tokens for use in the NER. Stanford NER is pre-configured to use this tokeniser's style of annotation

2.4.2. Brat rapid annotation tool

The tool chosen to annotate data in this project was brat. (Stenetorp et al. 2012) This tool was chosen as it is browser based, allowing for easy and collaborative annotation. In addition it is open source and allows anyone to host and manage their own annotation server.

After installing brat on a server, transferring the data to be annotated and configuring the entities & relations to be used for annotation, brat offers a simple and easy to use web interface with easy text selection for creating entities. Relations are created simply by dragging and dropping between entities, and all annotations are clearly visualised as can be seen in figure 2.1.

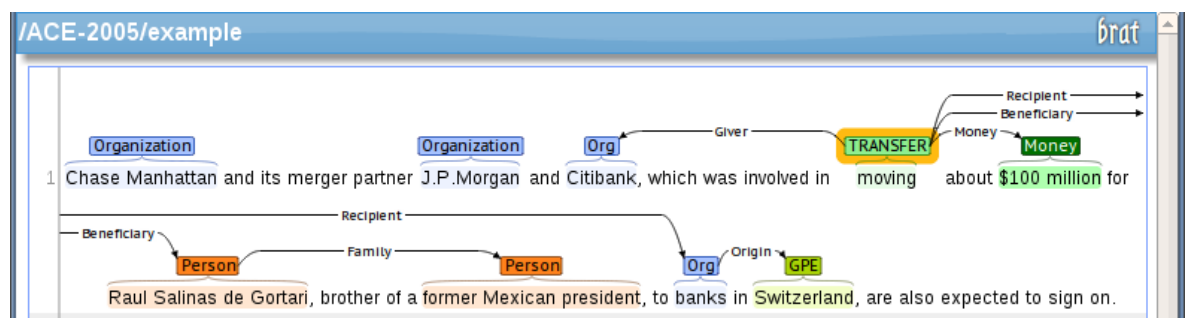


Figure 2.1 Entity annotation in brat

2.4.3. Scripts

The scripts used throughout this project for tasks such as data preprocessing and evaluating the NER were written in Python. Python was chosen as it is the most widely used language in machine learning applications, and consequently has many useful libraries for such applications as data preprocessing and metrics generation.

Initially Python scripts were used for data preprocessing, were libraries such as Pandas were useful for simplifying data manipulation, representing read files as easily modifiable data structures.

Scripts were later used to implement cross validation and bootstrap resampling methods for evaluating the NER. This was simplified by an easy to use library allowing the script to execute command line commands, allowing the NERs command line interface to be used and avoiding a complex interface between the NER, which is written in Java, and the Python scripts. Further libraries such as NumPy and SciPy were used to gather and generate the needed performance metrics.

Finally scripts using matplotlib, a visual graph generation library for Python, were used to generate the visualisations used to show the NERs performance and its correlation with dataset properties as seen in the evaluation.

3. State of the Art

In this chapter relevant state of the art work is presented and how it is relates to and influenced this work is described. This will cover prior natural language processing work in the domain of data governance, state of the art NER systems in related problem domains, and the governance model used in this dissertation.

3.1. Natural Language Processing in Data Governance

The only known investigation into the use of natural language processing in the area of Data Governance is a report written by Jaimin Shah at Collibra, “Utilizing Natural Language Processing and Artificial Intelligence to Identify Plausible Data Requests on Slack and Linking it to Collibra’s System of Record Tool DGC”.(Shah 2017) This report was not made publically available.

Shah investigated the use of machine learning classification models to identify messages containing potential data requests. The aim was to create a model which could label messages as either “contains a data governance request” or “doesn’t contain a data governance request”.

This problem was addressed using a binary classifier, and the report investigates the performance of a variety of classification algorithms in this task. These classifiers were trained and tested using an annotated dataset of approximately 200,000 messages, and found a support vector machine based classifier to have the best performance across all metrics, with a precision of 0.76 and recall of 0.75, as shown in table 3.1.

Table 3.1 Jaimin Shahs binary classifier evaluation

Classifier	Precision	Recall	F-1 Score
Multinomial Naive Bayes	0.7	0.7	0.7
Logistic Regression	0.74	0.74	0.74
Gaussian Naive Bayes	0.65	0.63	0.63
K-nearest Neighbour	0.74	0.61	0.58
Decision Tree Classifier	0.7	0.69	0.7
Support Vector Machine	0.76	0.75	0.75

This report was the only direct prior work in this field, as it investigated the potential of natural language processing in the identification of data governance entities and actions in chat messages. However the report was quite limited in scope, as the classifier was only binary, allowing no distinction between different data governance entities and actions. Additionally, the classifier labels entire messages, without identifying where in the message the data governance entity/action is.

Of note is the method of annotating messages, were the classifiers were first trained on a small hand-annotated dataset of ~200 messages, and the best classifier on this dataset (support vector machine) was used to find data request messages from the full dataset of 200,000 messages, finding 7,000 messages related to data governance. These messages were then reannotated manually, but the remaining 193,000 were left as negative examples. While this method was likely used to reduce annotation requirements and seems to have produced relatively good results, there are likely many instances of messages falsely annotated as not related to data governance. The use of the SVM classifier to filter these messages initially may also introduce a bias in its favor, as despite the reannotation of messages found to be data governance related, the unrelated messages were left unchanged from the SVMs predictions.

Despite these shortcomings, this report showed potential in the application of natural language processing in this problem domain, and the best performing classifier from this work was used in this project to filter the messages supplied by Collibra. This helped to reduce the number of irrelevant messages that needed to be annotated.

3.2. Related NER Applications

3.2.1. Domain Specific NER

While most work on named entity recognition has focused on identifying general entity types occurring in a wide variety of text, such as names, locations or organisations, there have also been numerous studies of NERs applied to more domain specific tasks, particularly in medical fields. Such NERs are trained on specialised data from their domain, and are generally tasked with identifying more specific entity types, e.g. Chemicals or Diseases. Due to the specialised domain upon which these NERs work, there are generally limited datasets available, and no annotated data exists for many domains.

Notable work in this area includes a paper exploring the use of a conditional random fields based NER on biomedical texts. (Wang et al. 2007) This paper seeks to recognise a number of named entities, such as proteins or DNAs. The paper explores a number of means of improving a NERs performance in this tasks as well as their impact on performance.

Explored are the set of features extractors used by the NER to select significant textual attributes that may indicate a words entity type. These features include whether the word has been seen before in the training data, capitalisation or certain characters present in a word, or word-level n-grams which record common word sequences.

It also explored how the dataset might be processed to improve the NERs performance, such as expanding abbreviations or splitting compound nouns of a chemicals name etc. into its component words.

Several of these methods of improving a NERs performance only resulted in a very minor performance improvement, and some dataset processing methods were not applicable to this dissertations dataset. Despite this the paper presented some interesting means of improving an NERs performance for a specific domain, and the chosen NER tool used in this dissertation, Stanford NER, has several of the feature extractors explored in this paper already implemented. Abbreviation expansion was also considered as a means of optimising the dataset, however the relative rarity of

abbreviations and the inconsistency of abbreviations in the text lead to this not being implemented.

Another notable paper in this area explored the use of deep neural networks in NERs to recognise disease names.(Tong et al. 2017) While the focus of this paper was on deep neural networks, it also explored means of data preprocessing such as sentence segmentation and word tokenisation which was used in this dissertations dataset.

Despite the use of a deep neural network and the extensive configuration and experimentation with the layers and connections within this network optimising its performance, the final model utilising this network only marginally outperformed a simpler CRF based model (89.6% F1 score vs 89.1%). This lead to the decision not experiment with neural networks in this work, as the significant complexity and time investment required in implementing a neural network would likely have led to negligible performance improvements.

There has been no published exploration of the effectiveness of a named entity recogniser of any kind in the domain of data governance.

3.2.2. Short and Informal Message NER

NERs are typically trained and used on formal and well formatted texts such as news articles and academic papers. The source of the dataset used in this dissertation however is an internal chat log. As this is a more informal environment, messages tend to be short and informal, meaning that correct capitalisation and sentence structure is not always used.

This will generally lead to a loss of accuracy for an NER, as many of the features used in predicting an entity type use elements such as part of speech or the capitalisation of a word. Additionally, words are much more likely to be misspelled or abbreviated in this environment, hurting the effectiveness of dictionary features relying on having seen the word in a training dataset.

There have been several papers investigating the application of NERs to informal texts, namely social media. A notable paper formally investigates what the main sources of error are in performing named entity recognition on tweets using state of

the art NER systems, and how these errors could be addressed in the future.(Derczynski et al. 2015)

This paper found that improper capitalisation had a particularly negative impact on NER performance, with a greater impact than slang or abbreviations. It investigated the use of part-of-speech tagging and normalisation to reduce the impact of noisiness in tweets, but ultimately found that precision and recall scores remained low using NER algorithms developed for use with formal texts.

The most notable paper in this area explores the use of word representations to improve the effectiveness of a NER in labelling Twitter messages.(Cherry and Guo 2015) This paper found that general NERs trained on formal texts performed very poorly in labelling tweets and sought to explore means of improving this performance.

The paper concluded that many typically used feature extractors reliant on textual properties mainly found in well edited text were almost useless in labelling this short, informal text. Brown clusters, an unsupervised method of grouping similar words together, and Word Vectors, a supervised method of doing the same, were explored as means of creating additional features used for entity prediction.

Due to the relatively small dataset of annotated tweets available, this paper also leveraged larger existing annotated datasets of formal text to increase the training dataset size, but weighted the informal tweet dataset more highly in the training process. The combination of these methods lead to a 20% improvement in F1 score on a test set of tweets vs a baseline NER trained on formal text.

This text lead to the exploration of the use of word vectors and Brown clusters in this dissertation, however it was decided not to use these methods in the NER, as quite a large dataset is required to get useful Brown cluster or word vector features.

Of note is that while the data governance chat data is informal, messages are generally more formally structured than social media text, with better adherence to sentence structure and correct spelling, so feature extractors reliant on properties present in formal texts are likely to perform better than in this paper, and the use of word representation methods would also likely have less of a positive impact on performance.

3.3. Collibra Data Governance Operating Model

The Collibra Data Governance Operating Model (CDGOM) is a meta-model developed by Collibra for the purpose of building a foundation upon which an organisation can build a model of their specific data stewardship and management activities.

The CDGOM was chosen for use in this dissertation as it is a very mature model that Collibra has successfully deployed across numerous organisations. It identifies the set of data governance entities that we wish to identify with the NER system, and in addition Collibra employees are familiar with and regularly use this model, meaning entities within a Collibra teams chat are more likely to directly conform to some entity from this model.

Other models of data governance exist, including an ISO standard for data governance, ISO 38500⁴. While this is an international standard, it has not been widely implemented and is hence immature in comparison to CDGOM which has been widely deployed. This results in Collibra's model being an overall more complete and useful model of an organisation's data governance assets, rules and relations.

CDGOM is broken down into 3 categories:

Structural Concepts, which addresses what is to be governed, i.e. defining the organization's data assets, their properties, and the relationships between them

Execution and Monitoring Concepts, which addresses how these assets are governed. This category defines the types of role users of the system have, which dictates their access rights, as well as rules for validating assets and defining their status or any issue they may have.

Organisational Concepts addresses who governs assets. It groups assets together into domains, which are controlled by communities containing users, domains and/or sub communities. Users within a community may access any assets in a domain owned by that community or any community that that community is a part of.

⁴ <https://www.iso.org/standard/62816.html>

3.3.1. Structural Concepts

Structural concepts will be explored in more detail as it is the most relevant category of the Collibra Data Governance Operating Model in this dissertation. This models the assets of an organisation as well as the data governance rules in place on them.

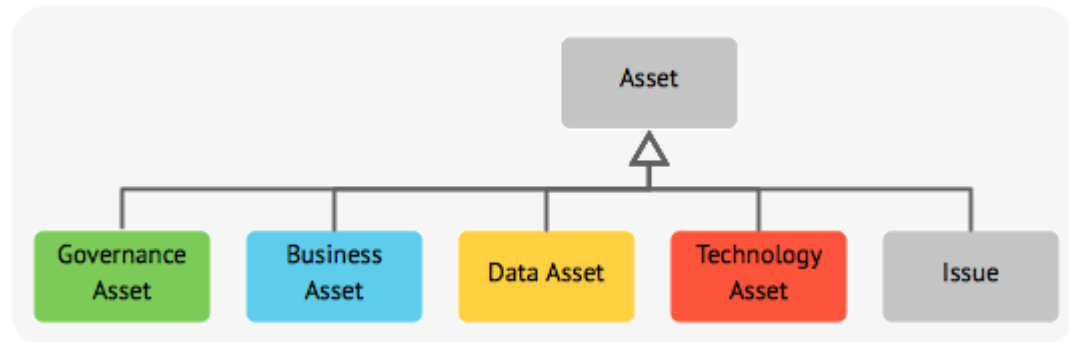


Figure 3.1 Collibra Data Governance Operating Models Main Asset Types

Structural Concepts defines 5 core asset types, shown above in figure 3.1. Each of these assets has several subtypes, i.e. Governance Asset has 2 subtypes, Policy and Rule. Rule in turn has 2 subtypes, Business Rule and Data Quality Rule.

Assets may have any number of attributes, which captures additional information about the asset. A set of attribute types are defined in the Structural Concepts document, with the type defining the class of information an attribute stores and what the information represents. A wide variety of attribute types are defined in the model, with some restricted to certain asset types or subtypes, i.e. the 'Purpose' attribute may only be applied to Governance Assets.

Also defined in structural concepts are relationships relating 2 entities. These relationships are directional, with each relationship having a reverse relation, i.e. if asset A governs asset B, asset B complies to asset A, as 'complies to' is defined as the reverse relation of 'governs'. Relations may also restrict the types of entities which may be used as the source and target of the relation. In the above example the 'governs' relation requires asset A to be a governance asset, while asset B may be any form of asset. Figure 3.2 shows the complete set of asset and relationship types.

4. System Design

This chapter describes the design of the systems used in this dissertation. It is broken down into 2 sections, dataset, which describes the source, preparation and annotation of the dataset used, and evaluation software, which details the scripts and tools used in evaluating the named entity recogniser's performance.

4.1. Dataset

This section will describe the dataset created for this dissertation, detailing its origin, the annotation scheme & process, and properties of the final annotated data.

4.1.1. Data Source

The raw data used to create this dataset was supplied by Collibra. It was a raw dump of messages from a Data Governance team at Collibra's slack chat.

Messages containing any particularly sensitive information were removed before being released, and the resulting data consisted of 7,022 messages and 300,000 words. However, a large proportion of these messages are not directly related to data governance. Due to this, filtering was performed to remove messages not related to data governance, as described in [section 4.1.3](#).

4.1.2. Annotation Scheme

The entities and relations chosen to be annotated in the dataset were based on the Asset Types from Collibra's data governance operating model detailed in [section 3.x](#). This approach was chosen due to Collibra employees' familiarity with this model, allowing easier annotation by Collibra in the future. Additionally, entities from within the chat are more likely to closely conform to this model as a result of this familiarity.

This approach lead to the initial entity types: Gov, Bus, Data, Tech and Issue, representing Governance Assets, Business Assets, Data Assets, Technology Assets and Issues respectively. However, upon annotating a sample of the dataset with this scheme, some tokens were found that did not fit under any of the entity types we had defined, but would be useful for the Named Entity Recogniser to label.

To address this, two additional entity types were taken from other sections of Collibra's data governance operating model. Role was used to label text representing a data governance role, such as an administrator or a domain expert, and User was used to label people interacting with the data governance model.

In addition, the Dmtask label was created to label text representing a data management task, such as upgrading or backing up a database. This was not a part of Collibra's model, but it was felt that it a useful entity to capture.

Relationships were also defined in the annotation scheme, allowing entities to be related to other entities. While these relations were not used elsewhere in this dissertation, the addition of relations to the dataset would allow it to be reused in future work such as a relation extractor, used to label relations between entities.

To avoid excessive complexity in the annotation process, only a small subset of the most commonly used relations defined in the Collibra Data Governance Operating Model were chosen to be a part of the annotation.

These are:

- 'Governed by', which can relate a governance policy to a data or tech asset it impacts. It must relate an entity of type Gov to an entity of any type.
- 'Uses', this is a relation between any two assets and specifies that one is dependent on the other for carrying out its function. This can be used to relate two entities of any type.
- 'Related to', this is a generic relation type that simply states that there is some relationship between any two assets. This can be used to relate two entities of any type.

As this annotation scheme is quite complex and it can be quite difficult to determine what tokens should be labelled as a given entity type and how various entities should be related, a document detailing this annotation process with examples and guidelines was created. This document is included as an appendix ([section 7.1](#)). The goal of this document was to unify the annotating style of any data annotators, allowing conflicting data to be minimised and increasing the potential performance of a NER trained on the dataset.

4.1.3. Data Preparation

Upon receiving the chat log from Collibra, it was found to contain some encoding errors, likely caused by the file being saved in a Windows encoding format from UTF-8. Many of these encoding errors were resolved simply by changing the files encoding, but a small number persisted. Due to their sparsity and the fact that it was unknown what characters these encoding errors should be replaced with, they were left in the data as they would have very little impact on the dataset.

As a large proportion of the messages in the data provided by Collibra were not directly related to data governance, it was necessary to filter unrelated messages out of the dataset to avoid unnecessary annotation work.

This was done using a SVM binary classifier previously developed with Collibra for the purpose of identifying messages related to data governance which is detailed in [section 3.1](#). This filtered the dataset from 7000 messages to around 800 messages.

While this classifier's performance was not optimal at 76% precision and 75% recall, it was good enough to ensure that the majority of the remaining messages (~600 out of 800 based on the classifier's precision) were related to data governance. Based on the classifier's recall score, around 270 messages related to data governance were also missed by the classifier, so the filtered data is not perfectly representative of all data governance related messages. However, the greater rate with which relevant messages could be annotated due to this filtering was needed, as limited time and resources were available for the annotation process.

Finally, to optimise the dataset for use with a Named Entity Recogniser, the data was tokenised using the Stanford Tokeniser, chosen as it was designed to be used with other Stanford NLP tools. This split the text into tokens, separating words from adjoining symbols that may obfuscate their meaning to the NER, i.e. '(word)' is tokenised to '(word)' or 'she's' is tokenised to 'she 's'. An exception to this was added for Slack's user references such as '<@U039FDLA>'. The tokeniser would normally split the angle brackets from the word, but in this case they may be valuable to the NER to identify the token as a user, so an exception was added to the tokeniser for this case.

4.1.4. Annotation Process

Annotation was carried out using a collaborative browser based tool, brat, described in [section 2.4.2](#). Much of this annotation was carried out by this projects supervisor, Rob Brennan, due to his greater familiarity with the nuances of Collibra's data governance model and hence the data annotation scheme that had been created.

The project's assistant supervisor Alfredo Maldonado and I also contributed to the annotation process, using the guidelines and examples created as well as inspecting some of Rob's annotations to keep the annotation style as consistent between all annotators as was possible.

The process of annotation took place over several weeks due to limited time available for annotation. Over this time the annotation style used evolved as common patterns and relations were found within the chat messages and annotators became more familiar with the entity types and relationships within the chat. This was also reflected in the annotation scheme, with examples and guideline being updated over this time.

This lead to some of the earlier annotated text being inconsistent with the majority of the annotated dataset. This inconsistency was later found to have had a negative impact on the performance of the NER trained on this data, and removing the earliest ~10% of annotated data resulted in a noticeable improvement to the NERs performance (approximately 4% across all metrics), showing the negative impact that inconsistent annotation can have on an NERs accuracy.

While this early annotation was the most inconsistent with the later annotation style, lesser inconsistencies in annotation style over time and between annotators still existed. These could not be addressed as there was not enough time to reannotate the dataset, and removing more of the annotated data would have had too much of a negative impact on the datasets size. In future work, stricter annotation guidelines may help further minimise these inconsistencies, and annotators should practice annotating a sample of the dataset before contributing the datasets annotation.

4.1.5. Dataset Properties

The final annotated dataset consisted of 4749 tokens, with 3011 of these being annotated as 'O', or not an entity. Properties of each entity type in the final dataset are shown in table 4.1 below, with visualisations of this data in figures 4.1 and 4.2.

	Annotated Words	Unique Words	Length Mean	Length Standard Deviation
Bus	196	141	1.6752	1.2665
Data	503	217	1.7964	1.4459
Dmtask	144	93	1.1707	0.5054
Gov	182	114	3.7143	3.2262
Issue	310	175	3.4444	2.3623
Role	14	9	1.5556	0.8315
Tech	236	129	1.4937	0.9727
User	153	44	3.1875	1.8892
Full Dataset	1738	922	1.8242	1.7729

Table 4.1 Properties of each entity type in the dataset

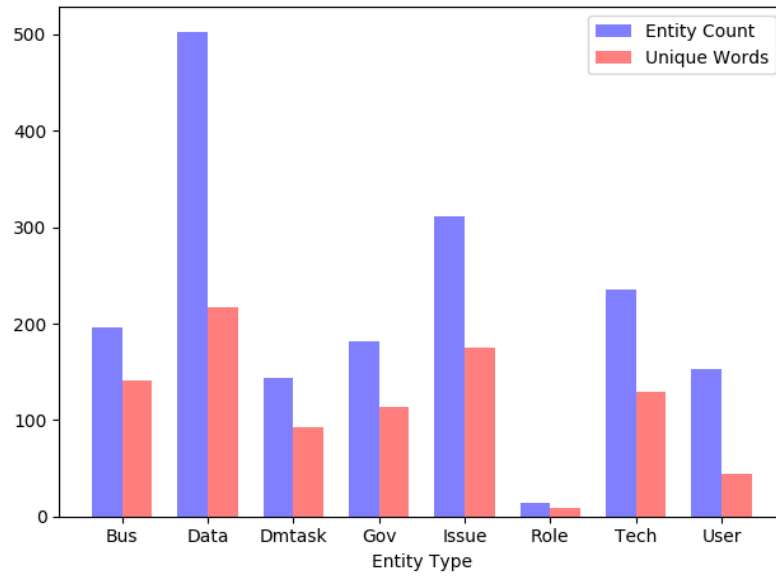


Figure 4.1 Word count & Unique word count of each entity type

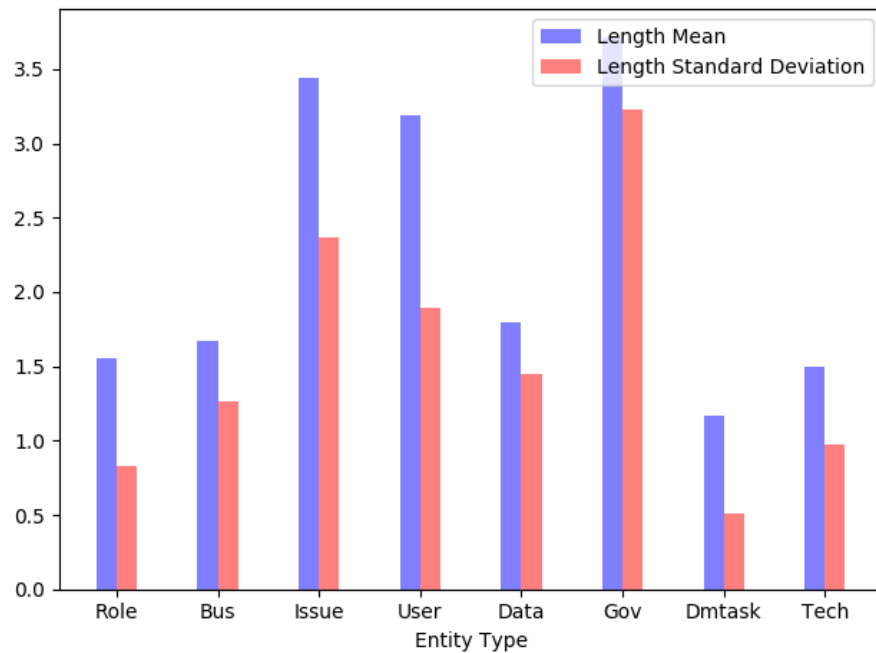


Figure 4.2 Mean and Standard Deviation of the length of each entity type

4.2. Experimentation and Evaluation Software

This section will explore in detail the processes used in training, testing and gathering metrics on the named entity recogniser, as well as generation of useful figures from these metrics. The tools, scripts and libraries used, and how they are utilised is covered in detail.

4.2.1. Initial NER System Experimentation

Prior to the dataset being annotated, some initial experiments with the NER system were performed. This consisted of training the Stanford NER system on a very small set of around 10 hand-annotated messages and testing several unannotated messages. This was done via Stanford NER's command line interface (CLI).

The performance of the NER system in this initial experimentation was of course very poor, though it was capable of identifying some common tokens such as 'data' or user identifiers e.g. '<@U1234567>'. This showed the promise of the NER systems labelling ability for some entities even on a tiny initial training dataset. However, this experimentation was done primarily to gain familiarity with the Stanford NER tool and the data format it received and output.

4.2.2. Automation Scripts

In order to have a uniform testing environment for evaluating the NER's performance variations with changes to either its configuration or the dataset, a Python script was written to run a range of testing methods on the NER.

The first iteration of the script had a simple split testing function, splitting the dataset into two sections, training on one section and testing on the other. This was done by creating two temporary files, each containing one section, and using the Python subprocess library to train and test Stanford NER on these files via the CLI. The function interfacing the Python script with the Stanford NER CLI is shown in figure 4.3.

```

def run_ner(train, test, config, name=0):

    with open("trainfile.txt", "w") as output:
        output.writelines(train)
    with open("testfile.txt", "w") as output:
        output.writelines(test)

    # Train and test via command line
    nertestcommand = "java -cp stanford-ner/stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -
trainFile trainfile.txt -prop {} -testFile testfile.txt".format(config)
    out = subprocess.run(nertestcommand, stdout=subprocess.PIPE, stderr=subprocess.PIPE, shell=True)

    # Write results to file
    with open("NEROutput/NEROutput{}.tok".format(name), "w") as output:
        output.write(str(out.stdout, 'utf-8'))

    out = str(out.stderr, 'utf-8')
    pattern = r'Totals\t(.....)\t(.....)\t(.....)'
    # Extract results from stderr with regex
    reg = re.search(pattern, out, re.M|re.I)
    p = float(reg.group(1))
    r = float(reg.group(2))
    return (p, r, f_beta(p, r))

```

Figure 4.3 Python function to train and test Stanford NER CLI

Stanford NER's CLI prints overall precision, recall and F-1 scores to stderr as a part of its output, so the script initially captured these performance metrics directly from this output using regular expressions.

Following this initial implementation, the script was expanded with a cross validation function. This called the split testing function for each fold and aggregated the resulting set of performance metrics. Bootstrap was also implemented as an experimental function in order to compare it with cross validation. It was decided to proceed with cross validation rather than bootstrap for reasons detailed in [section 2.2.2](#), so the bootstrap function was later scrapped and became nonfunctional as the script was expanded upon.

A testing configuration file, shown in figure 4.4, was used to select the testing method to be used as well as the parameters for the test methods, e.g. number of folds for cross validation or the train/test split percentage for split testing. This file also defined the set of annotated data files to be used. The location of Stanford NER's

configuration file was made configurable here, with the option to give multiple NER configuration files in which case the testing would be repeated with each different NER configuration file. This was implemented to speed up experimentation with varying configurations of the NER's features.

```
[MAIN]

test_type = cross_validation
# test_type = split
# test_type = bootstrap

# cross_validation only
folds = 4

# split only
split = 70

f-beta = 1

# add additional data files / configs separated by newline
test_data: Datasets/v5/ner_data/ner-crf-training-data.tsv

ner_configs: config.prop
```

Figure 4.4 Evaluation script configuration file

4.2.3. Metrics

While Stanford's CLI outputs precision, recall and F1 scores for the dataset as a whole, it does not give this score for each entity type, nor does it allow the F-value to be varied from 1.

In order to gather these more detailed metrics for analysis, first F-scores were no longer taken from the Stanford NER CLI output and instead calculated from the precision and recall values, with the F-value configurable in the configuration file.

The CLI did however output each token in the test dataset with the entity type it was annotated with and the entity type that was predicted by the NER. The evaluation script was updated to write this data to files, allowing the metrics script to compute detailed metrics from this data.

Functions were then written to calculate the precision, recall and F-scores of the NER for each entity type, and additionally gather more detailed statistics on properties of each entity type in the dataset. An excerpt of the scripts output for one entity type is shown in figure 4.5.

```
Data

Annotated: 503
Unique Word Count: 217
Average Length: 1.7964285714285715
Length Standard Deviation: 1.4459258385588953

Predicted: 404
Precision: 0.6138613861386139
Recall: 0.49304174950298213
F-1.0: 0.5468577728776185
```

Figure 4.5 Metrics script output excerpt

4.2.4. NER Experimentation

With the testing and metrics scripts completed and a larger dataset, some experimentation was performed with NER, namely varying settings with the NER's configuration file. This allowed the enabling, disabling, and tweaking of a range of feature extractors built into Stanford NER. The full set of options that can be configured in Stanford NER's configuration file can be found in their documentation⁶.

A variety of these options were trialed used the evaluation scripts, such as using the lemma (dictionary form) of words, or varying the size and use of character-level n-grams for using word prefixes or suffixes as features. Many configurations were trialed, but ultimately the overall best performing configuration file was very similar to the default supplied with the NER, demonstrating a high level of optimisation of the Stanford NER tool. The final configuration file of NER is shown in figure 4.6.

⁶ <https://nlp.stanford.edu/nlp/javadoc/javanlp/edu/stanford/nlp/ie/NERFeatureFactory.html>

```
serializeTo = ner-model.ser.gz
map = word=0,answer=1

useClassFeature=true
useWord=true
useTypeSeqs=true
useTypeSeqs2=true
useTypeSequences=true
wordShape=chris2useLC
useNGrams=true
noMidNGrams=true
useDisjunctive=true
usePrev=true
useNext=true
maxNGramLeng=8
useSequences=true
usePrevSequences=true
maxLeft=1
```

Figure 4.6 The best performing Stanford NER configuration

4.2.5. Plotting

In order to generate plots needed for the evaluation section of this dissertation, a separate script was written which used the output of the metrics script to generate a set of bar charts and scatter plots to display and correlate performance metrics of the NER and properties of each entity type.

To work with the data generated by the metrics script in a separate script, the metrics were structured in a python dictionary which was then serialised and written to a file using Python's pickle module⁷. This file was then read and de-serialised to the original dictionary in the graphing script.

Graphs were generated using Matplotlib⁸, a plotting library for python capable of generating a wide variety of high quality graphs and figures.

The first graph created was a bar chart showing the NER's precision, recall and F-1 scores for each entity type, followed by charts representing various properties of the

⁷ <https://docs.python.org/3/library/pickle.html>

⁸ <https://matplotlib.org/>

entity types in the dataset. They can be seen in figures 4.1, 4.2 and 5.2. These graphs were quite easy to generate as it is a common use case of Matplotlib and many examples exist.

Scatter plots were required for correlating the NER's performance with properties of the dataset for each entity type. These plots can be seen in figures 5.3, 5.4, 5.5 and 5.6 in chapter 5. While scatter plots were documented by Matplotlib, there were much less examples and documentation was less complete than bar charts. This resulted in much more required trial-and-error experimentation to generate the desired scatter plots.

In order to better visually represent the correlation between a given metric of the NER's performance and a given property of the dataset, a line of best fit was added to each scatter plot. This line was calculated using `numpy.polyfit`, from the Python library NumPy. This function calculated the least squares polynomial fit of the points in each scatter plot.

A numeric representation of the correlation in the scatter plots was also calculated using `numpy.corrcoef`, which returns the Pearson correlation coefficients for each plot. All of the scatter plots used in the evaluation were generated with the code show in figure 4.7.


```

met = ['precision', 'recall', 'f-measure']
stat = ['annotated', 'avg_len', 'len_std', 'uni']

for me in met:
    for st in stat:
        avg_len, f = [], []
        for k in stats:
            # Special case for frequency of unique words
            if st == 'uni':
                avg_len.append(stats[k]['unique_words']/stats[k]['annotated'])
            else:
                avg_len.append(stats[k][st])
            f.append(stats[k][me])

        avg_len = np.array(avg_len)
        # Least squares polynomial fit
        m, b = np.polyfit(avg_len, f, 1)
        # Pearson correlation coefficient
        corr = np.corrcoef(avg_len, f)
        print("Correlation of {} {}: {}".format(st, me, corr))

        fig, ax = plt.subplots()

        ax.plot(avg_len, f, '.')
        ax.plot(avg_len, m*avg_len + b, '-')

        ax.set_xlabel(st)
        ax.set_ylabel(me)
        fig.tight_layout()
        fig.savefig("graphs/{}_{}.png".format(st, me))

```

Figure 4.7 Scatter plot generation code

5. Evaluation

5.1. Introduction

This chapter seeks to evaluate the effectiveness of the developed Named Entity Recognition system with respect to the research question, 'To what extent can Named Entity Recognition be used to extract data governance information and actions from a chat channel'. This evaluation focuses on an experiment determining the labelling accuracy of the NER system on a representative annotated dataset containing data governance entities.

As shall be seen in the results, the accuracy of the NER system varies according to the entity type it seeks to predict. We hypothesise that some entity types are more difficult to predict than others and this difficulty depends on the complexity of the entity type.

Detailed in this chapter are the hypothesis of the experiment carried out, the methodology of the experiment, the results of the experiment, and an analysis and conclusion based on these.

5.2. Hypothesis

The hypothesis for this experiment is:

The accuracy of a NER system for a given entity type is correlated with the complexity of the entity type and its number of occurrences.

In the context of this dissertation the complexity of an entity type is defined as the number of unique words of that type relative to the total number of words of that type, as well as the mean and standard deviation of the length of the entities occurrences (in words).

The reason these measures of complexity were selected is that NER systems generally have more difficulty identifying tokens that they have not previously seen, particularly when it is trained on a small dataset as in this case. Additionally, due to performance constraints, NERs can only take into account a limited context of

neighbouring tokens for identification, making complex context-dependent labelling a difficult task. Conversely, entities that tend to be longer (in words) may give the NER more context information with which it can label tokens, if the entity structure is not too complex or diverse.

To test this hypothesis, statistics will be gathered on the accuracy of the NER for each entity type and these will be compared to the properties of that entity type hypothesised to affect the NERs accuracy. The statistics measuring NER accuracy will be precision, recall, and F-1 scores, and the properties of the entity types used will be number of annotated instances, the type-token ratio, the mean length in words, and the standard deviation of that length.

This will also give a better understanding of how the annotation scheme affects the performance of the NER and how it could be altered to improve the performance of the NER, such as removing uncommon or overly complex entity types from the scheme or changing annotation guidelines to simplify entities and minimise the factors adversely affecting the NERs performance.

5.3. Methodology

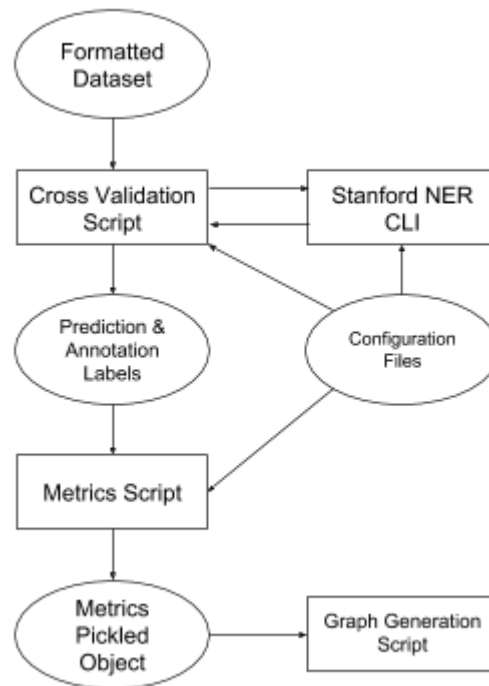


Figure 5.1: Experimental Protocol

Figure 5.1 shows a high level diagram of the resources and tools used in the evaluation process. Rectangles denote scripts or programs called, while ovals represent data produced or used by them.

Configuration files for Stanford NER set various hyperparameters and file format properties, while the scripts configuration file sets the number of folds to be used for cross validation and the metrics to be collected.

The cross validation script takes the annotated dataset and splits it into a training and test set for each fold before calling the Stanford Named Entity Recogniser Command Line Interface to train and test on each fold. The annotated and predicted labels for each fold are then joined and output as a file.

This file is then used by the metrics script to calculate the precision, recall and F-1 scores of the NER on each entity type, as well as properties of the entity type from

the dataset such as the number of occurrences or its average length. These are output as a Python dictionary serialized and written to a file, as well as being printed to terminal.

The saved metrics file is then used by the graph generation script to produce visualisations of the metrics.

5.3.1. Cross Validation Approach

In order to gather the required results, it was necessary to split our dataset, described in [Section 4.2.](#), into training and testing sets. Due to the relatively small size of the dataset, it was decided to use cross-validation as the resampling method as it allows all data point in the dataset to be used for training and testing.

Resampling methods are explored in more detail in [Section 2.2.2.](#)

The performance metrics used to benchmark the NERs performance on the test set are precision, recall and F-1 score, explored in [section 2.2.1.](#)

While 10-fold cross-validation with stratification is generally considered optimal for reducing bias and variance for accuracy estimation (Forman & Scholz 2010), 4-fold cross-validation was chosen due to the small dataset size. If 10-fold CV was used, the test portion would be quite small, leading to test sets which may not be representative of the dataset. There was also an increase in variance when using 10-fold CV. This can be seen in table 6.1, where a comparison of the mean and standard deviation of the NERs performance metrics across 4, 5 and 10 fold cross validations shows 10 fold CVs standard deviation in F-1 score is approximately 40% higher than 4 or 5 fold CV. These performance metrics are overall results taken across all entity types.

Additionally, the context dependent nature of the NER's predictions make stratification difficult to use to evenly distribute tokens, particularly given the datasets small size which made an even distribution of entity types amongst the folds while keeping messages intact impossible.

Table 6.1: Mean and Standard Deviation of performance metrics across folds of 4, 5 and 10-fold Cross-Validation

	Precision	Recall	F-1 Score
4-fold CV	0.4516±0.01255	0.2487±0.08224	0.3143±0.06584
5-fold CV	0.4334±0.04266	0.2444±0.06982	0.3092±0.06380
10-fold CV	0.4540±0.09512	0.2612±0.09045	0.3262±0.08908

To implement cross validation, a python script was used to split the dataset into the training and test segments for each fold, and call the Stanford NER Command Line Interface to train and test on these segments for each fold. Stanford NER CLI then produced precision and recall metrics which we then macro-averaged over the folds and used to produce our F-measure.

However, as the Stanford NER CLI did not produce performance metrics for each entity type, another script was required to go through the output from the test set of each fold to calculate the precision, recall and F-measure for each entity type across all folds.

This evaluation was iterated upon, evaluating different hyperparameters of the NER such as the use of character-level n-grams. The evaluation results displayed represent the best performing hyperparameter settings found. More technical detail on the evaluation methodology can be found in [section 4.2](#).

5.4. Results

Table 6.2: 4-Fold Cross-Validation Results

Performance Metric	Precision	Recall	F-1 Score
Bus	0.3611	0.0663	0.1121
Data	0.6139	0.4930	0.5469
Dmtask	0.4918	0.2083	0.2927
Gov	0.3684	0.0385	0.0697
Issue	0.3889	0.0675	0.1151
Role	N/A	0	N/A
Tech	0.6423	0.3347	0.4401
User	0.8831	0.8889	0.8860
Overall	0.4516	0.2487	0.3143

The overall results and results for each entity type from 4-fold cross-validation can be seen table 6.2. Across all metrics, the NER performed by far the best on the User entity type. It also performed well on both Data and Tech, achieving relatively high precision, but with worse performance in Recall. Aside from Role, which was never predicted due to its rarity in the dataset, the NER performed the worst on the Gov, Issue and Bus entity types, all of which had very low recall and relatively low precision. With the exception of User, all entity types had notably higher precision than recall.

A visual comparison of the NER's performance by entity type is found in figure 5.2.

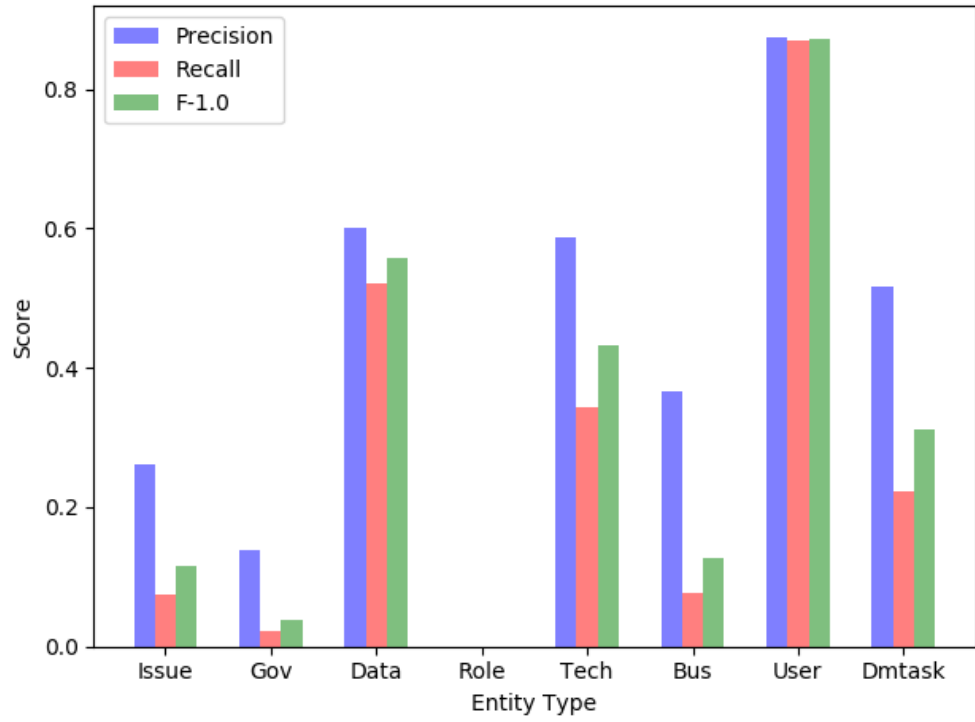


Figure 5.2 Results from 4-fold Validation for each entity type

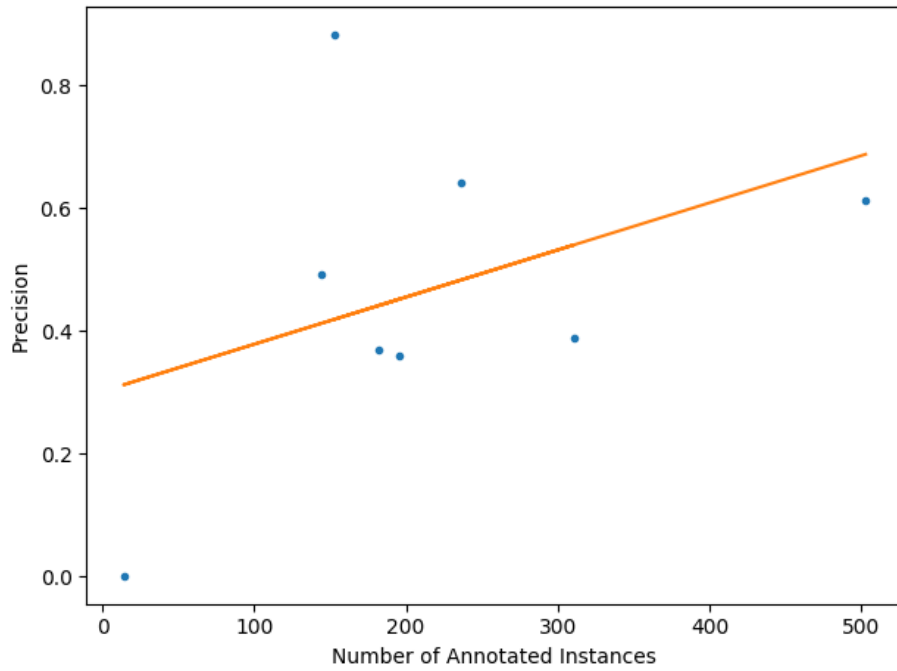
5.5. Analysis

In this section the results of the experiment will be analysed and the NERs performance on each entity type correlated with properties of that entity type in the dataset in order to test the experiments hypothesis. These properties are the number of annotated instances of an entity type, the type-token ratio of an entity type, the mean length of an entity type, and the standard deviation of an entity type's length.

The overall results from 4-fold cross-validation show a precision of 0.4516, a recall of 0.2487 and an F-1 score of 0.3143, however as can be seen in figure 5.2, there is a large difference in the performance of the NER between different entity types, particularly in recall. This high disparity in the NERs performance between entity types proves useful for evaluating the experiments hypothesis, as a range of values in the NERs performance is required for a correlation to be found, and a larger range of values may make any correlation found more statistically significant.

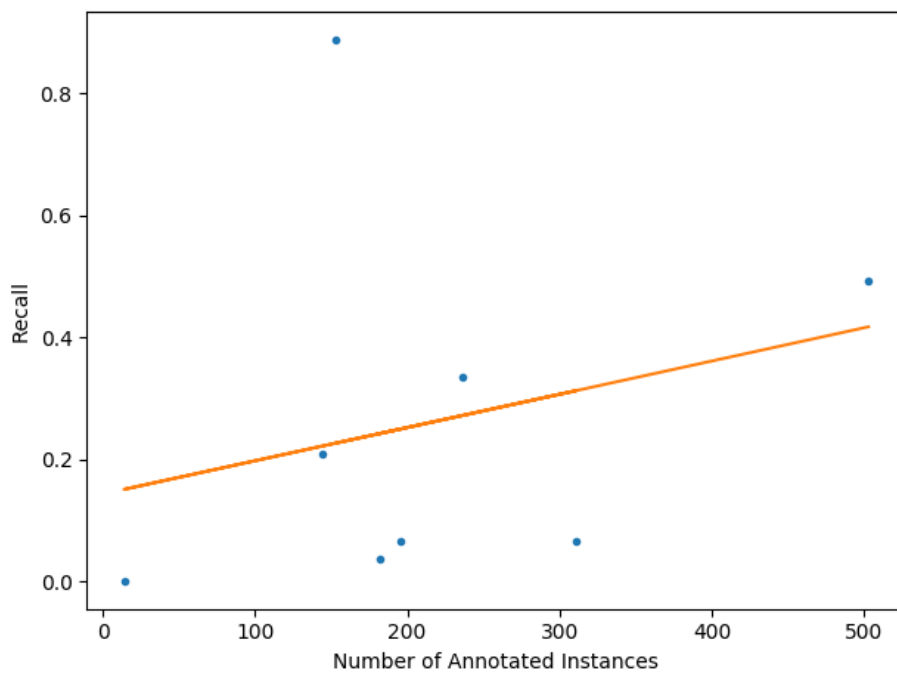
In the following 4 subsections the correlations between properties of each entity type from the dataset and the NERs accuracy on that entity type will be evaluated in order to explore how each of these properties are related to the performance of the NER.

5.5.1. Impact of Number of Instances Annotated on NER Accuracy



Pearson
Correlation
Coefficient

0.35



0.29

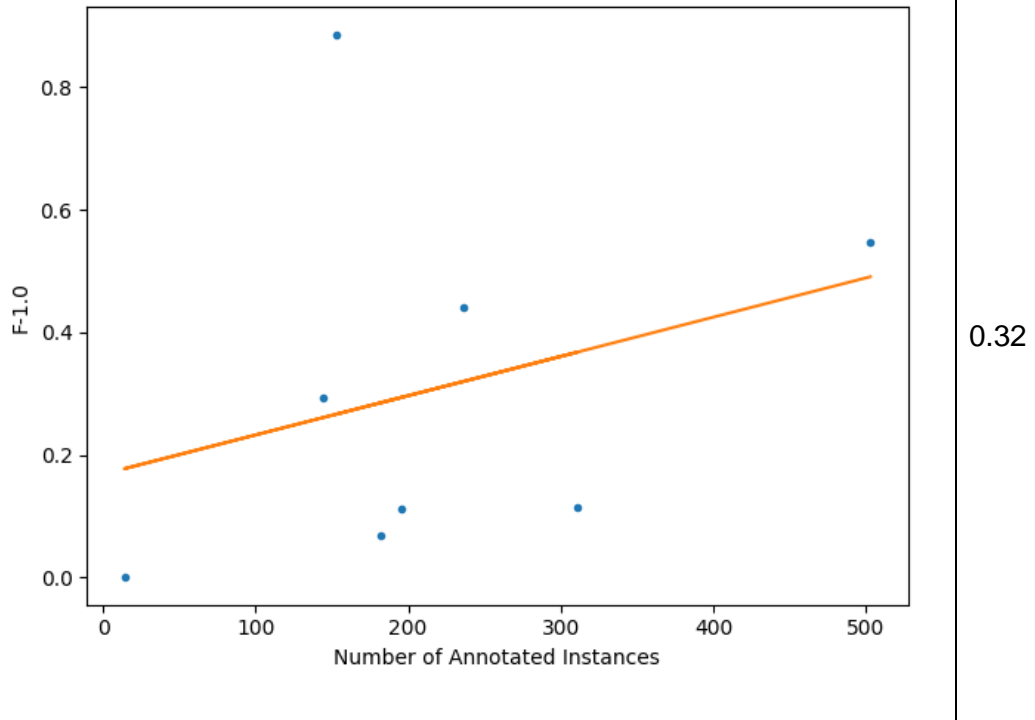
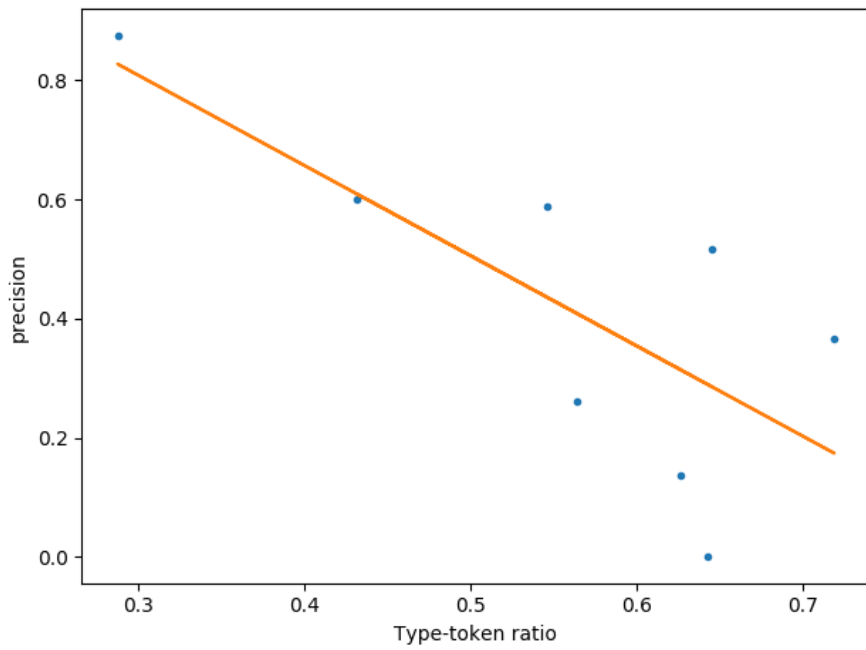


Figure 5.3 Correlation between number of annotated instances and performance of an entity type

From figure 5.3 we can see that there is a noticeable correlation between the number of annotated instances of an entity type and the NER systems accuracy on those entity types. The correlation is similar across precision and recall, indicating that an increased training size for an entity type may have a similar positive impact on both measures of the NER systems performance. Note the line of best fit shown in the figure is a least squares polynomial fit.

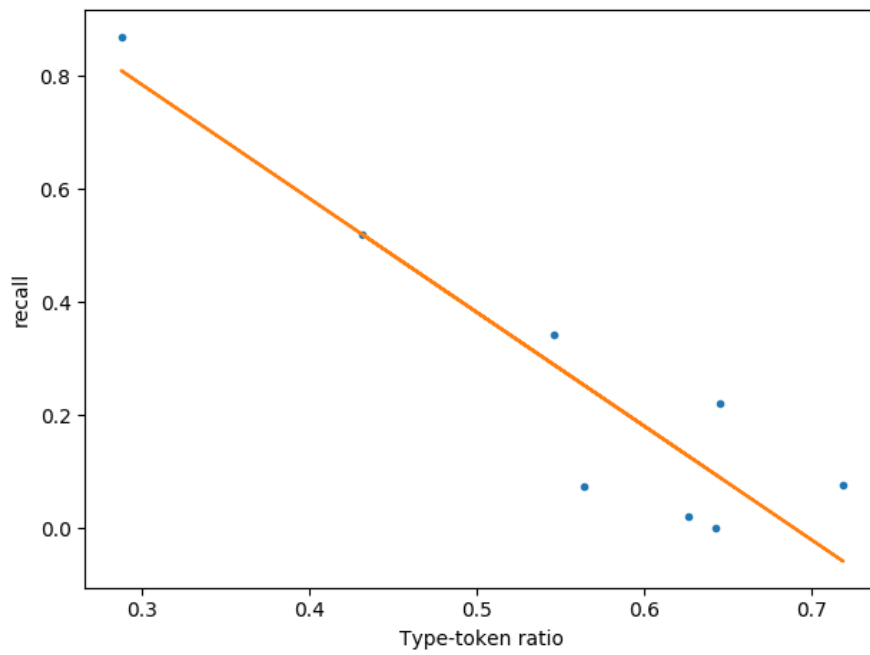
While the interpretation of the strength of a correlation is very much context dependent, i.e. a correlation of 0.9 may be considered weak if verifying a physical law with accurate measurements, but considered very strong in an area such as social science where there may be many factors complicating the data, the correlation seen here is certainly not particularly high, perhaps indicating that training dataset size is not the most influential factor on an NERs performance for an entity type.

5.5.2. Impact of Type-Token Ratio on NER Accuracy



Pearson
Correlation
Coefficient

-0.74



-0.92

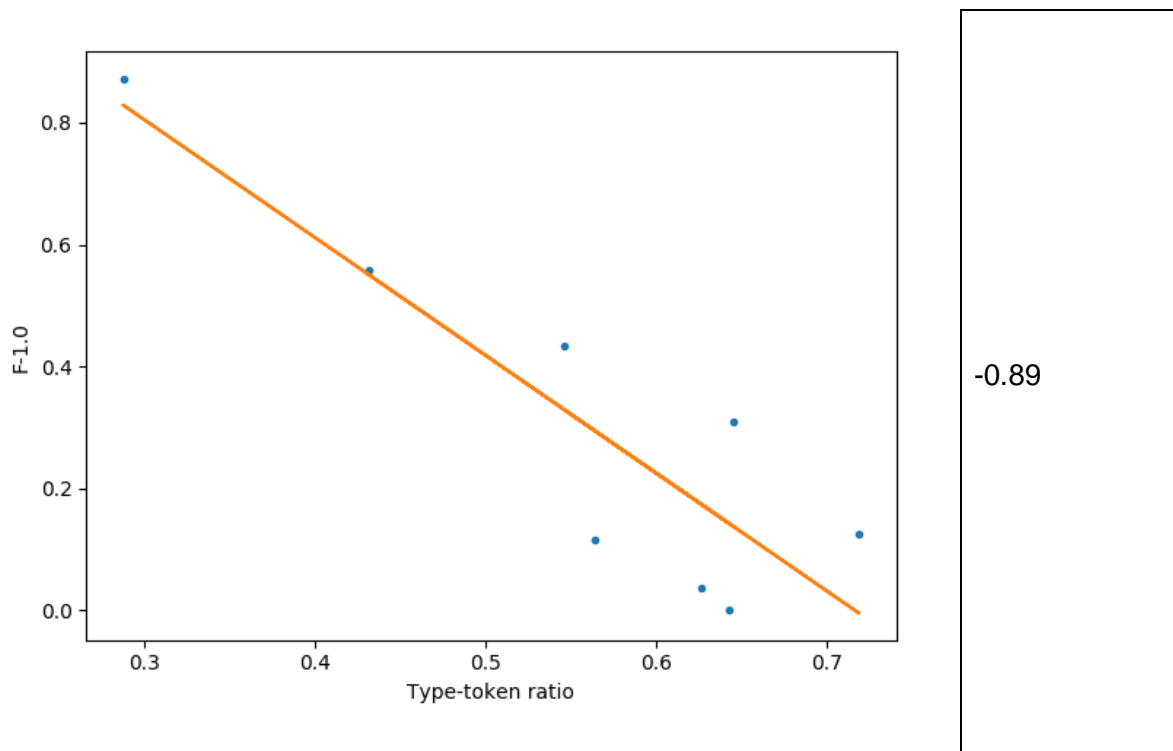


Figure 5.4 Correlation between type-token ratio and performance of an entity type

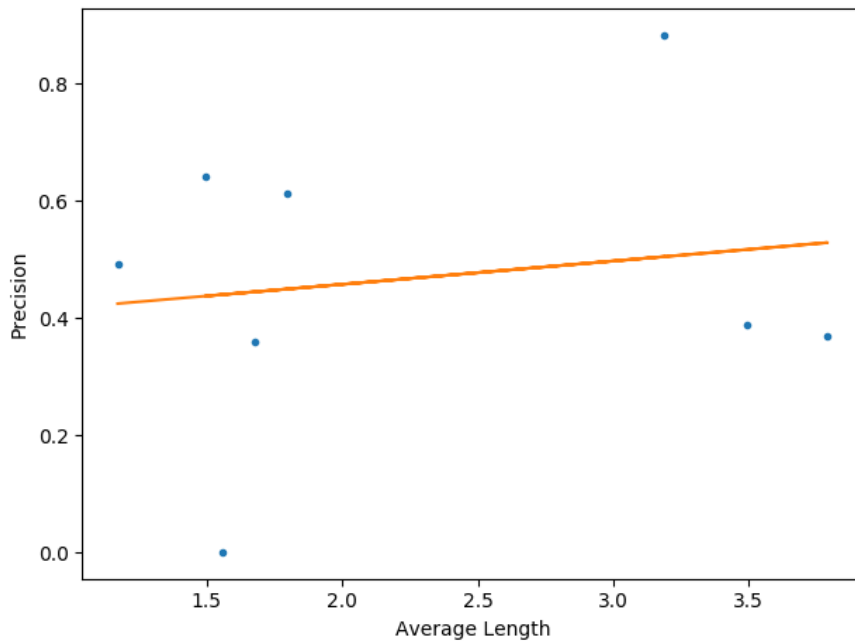
Comparing the frequency of unique words (number of unique words of an entity type divided by the total number of words of that entity type) to the NERs performance on that entity type in figure 5.4, a very strong negative correlation can be seen, particularly in recall.

This shows the high negative impact that high word diversity in an entity type can have on an NER systems performance, with this high negative correlation suggesting that it may have an even greater impact on NER performance than the number of instances of an entity type. The greater negative correlation in the recall metric may be explained by words unseen in the training set occurring less frequently in the test set when there is a lower type-token ratio, lowering the probability of the NER encountering a previously unseen word and producing a false negative.

This result was expected, as NERs, as with most other machine learning algorithms, perform best on more homogenous datasets. Additionally the use of a dictionary feature storing previously encountered words in this NER further improves this NERs performance on previously seen words. However the strength of the correlation between the NERs performance and this measure of complexity was unexpectedly

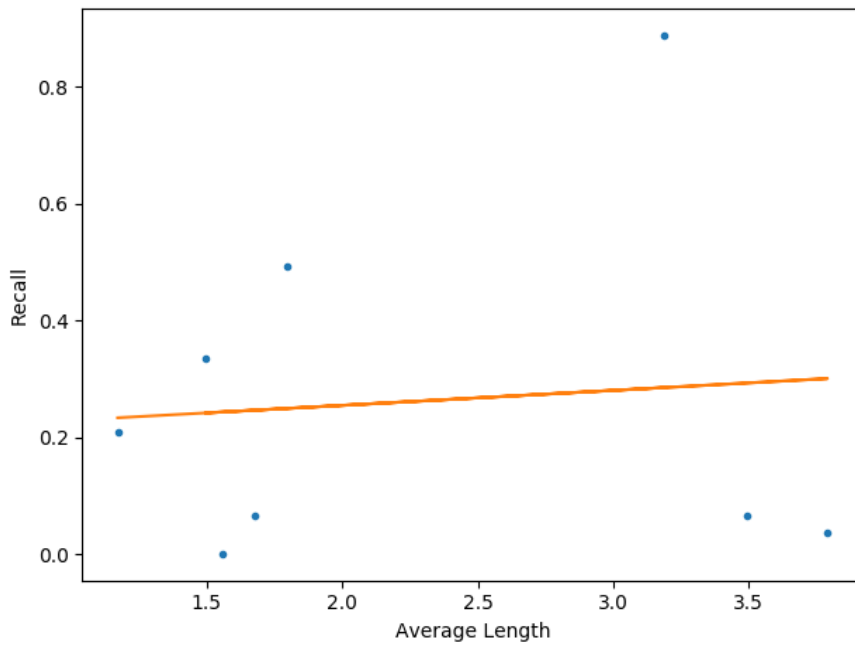
high, showing the type-token ratio to be a very good performance indicator for a NERs performance on an entity type.

5.5.3. Impact of Mean Entity Length on NER Accuracy



Pearson
Correlation
Coefficient

-0.08



0.07

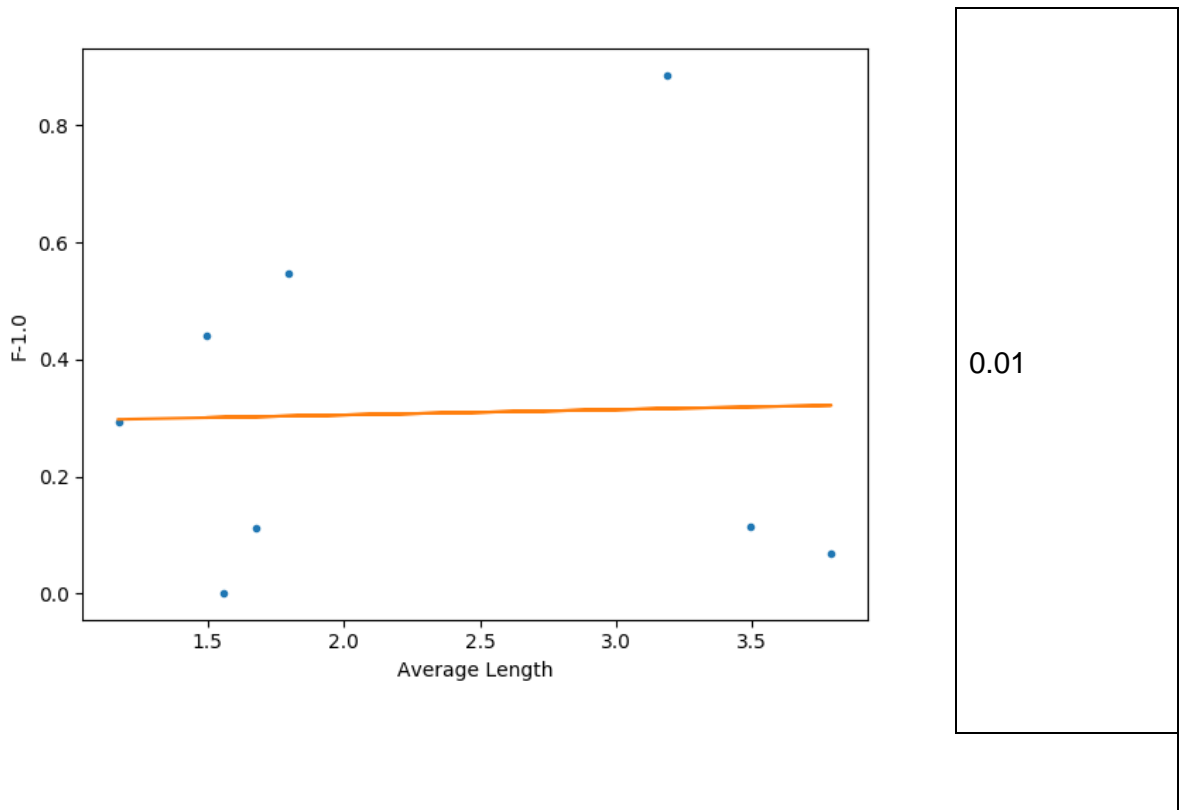


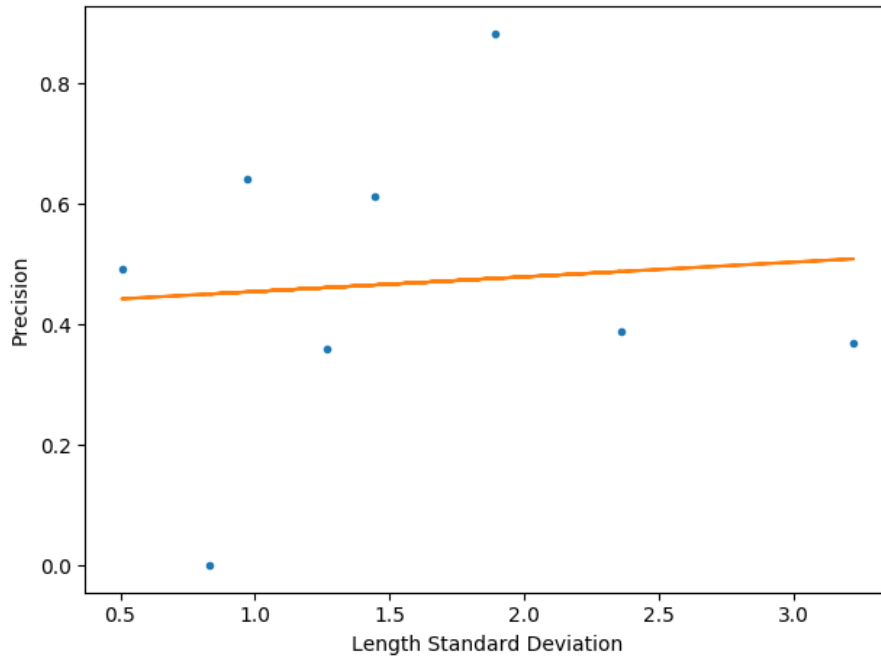
Figure 5.5 Correlation between entities mean length and performance of an entity type

Figure 5.5 shows the correlation between the mean length of an entity type (in words) and the performance of the NER on that entity type. No noticeable correlation was found between the mean length and any of the 3 performance metrics, indicating that it has little impact on the NERs performance for this dataset.

However there are several factors that could be affecting this correlation that may vary across dataset. First, an increase in the mean length of an entity type could have both positive and negative implications for the NER, where a long chain of tokens may give the NER additional context for labelling tokens, while conversely the NER may not be able to take into account the full chain of tokens if it is too long, as runtime of the NER increases exponentially with an increase in context window size. The impact of these 2 factors will likely vary depending on the nature of the text in the dataset, which could lead to a greater correlation in some datasets. For the dataset used in this dissertation, Gov and Issue entities had the longest mean length, and the NER system performed quite poorly on both, with recall being particularly bad. This may be in part explained by common words such as ‘the’ being included in these long annotations, and the NER labelling them as ‘O’ as a result.

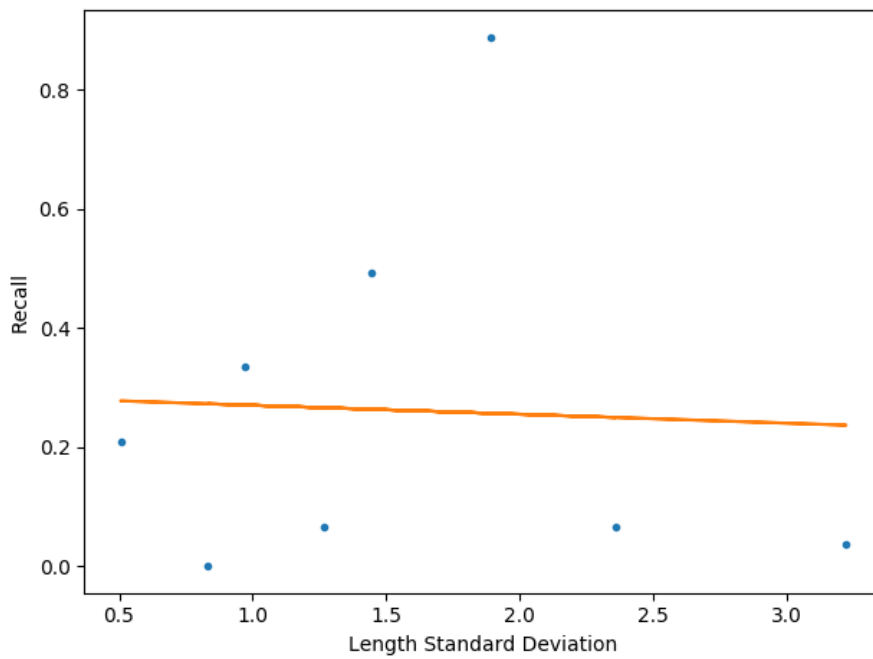
Additionally, the mean length of the entity types range from 1 to 4 in this dataset, which is not a high enough mean length for the NER to have difficulties taking into account the full context of the set of tokens. Datasets with entity types with longer mean lengths may see a greater negative correlation between NER performance and mean entity length, however this case would be quite rare, as most multi-word entity types in English are not longer than 3 words.

5.5.4. Impact of Standard Deviation of Entity Length on NER Accuracy



Pearson
Correlation
Coefficient

-0.19



-0.07

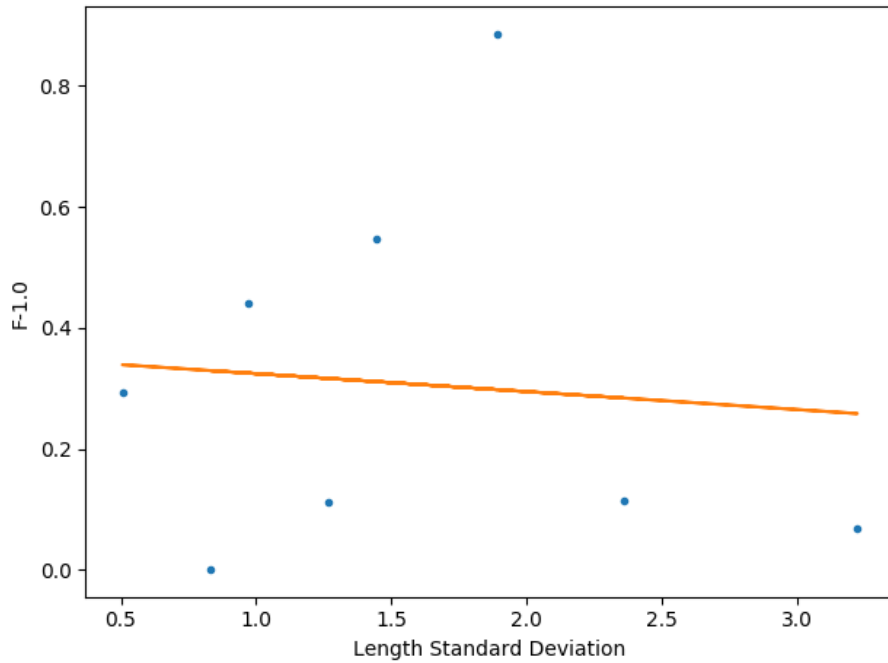


Figure 5.6 Correlation between standard deviation of entity lengths and performance of an entity type

Figure 5.6 shows the correlation between the standard deviation of the length of an entity type and the NER system's performance on that entity type. A slight negative correlation was found between the standard deviation and NER system's performance, indicating that entity types with a high variability in length may be harder for an NER system to predict, perhaps due to the more variant contextual information that the NER system has to infer from.

However the relatively weak correlation in conjunction with the small sample size makes this conclusion indefinite as other unrelated factors may give rise to this correlation. This is a similar result to mean length, indicating that the word length of an entity type in general has little impact on the NER systems performance.

5.5.5. Summary

The preceding four sections evaluated the correlations between the NER system's performance and dataset properties hypothesised to affect Named Entity Recognition performance.

A very strong negative correlation was found between the type-token ratio of an entity type and the NER system's performance, indicating that an entity types lexical diversity is closely related to an NER system's performance, even more so than the number of annotated instances in the case of this dataset. While there was only eight data points (the number of entity types) for this correlation, the correlations strength reveals a very clear relation between the 2 variables.

The correlation between the mean and standard deviation of an entity type's length and the NER system's performance on those entity types was explored as it was thought that these factors might have an effect on the quality of contextual information that the NER uses to predict labels, given the limited range of words it can use as context. However, no correlation was found between the NER system's performance and an entity types mean length, and only a slight correlation was found between the standard deviation of an entity type's length and the NERs precision performance, indicating that these properties of the data have little to no effect on the NER system's performance.

With respect to the experiments hypothesis, 'The accuracy of a NER system for a given entity type is correlated with the complexity of the entity type and its number of occurrences', of the four properties hypothesised to be correlated with a NER system's performance, two were found to be correlated: The type-token ratio of an entity type, which was found to be strongly negatively correlated, and the number of instances of an entity type, which was more weakly correlated. Conversely, no correlation was found between the mean length of an entity type and the NERs performance, and only a slight correlation was observed between the standard deviation of an entity type's length and the NER system's performance.

Overall this experiment has shown a notable correlation to exist between the NERs performance and two of the four properties hypothesised to influence that performance, proving the hypothesis partially correct.

5.6. Conclusion

The most important finding in this evaluation was the strength of the correlation between the type-token ratio of an entity type and the NER system's performance on that entity type. This may indicate potential to increase an NER system's performance by changing the annotated dataset, aside from the typical approach of increasing the dataset size. It may be possible to revise the annotation scheme and guidelines to create a less diverse set of entities, for instance not annotating words indirectly referring to an entity, and removing very rarely occurring entities, such as Role in the evaluation dataset, entirely.

This would result in a less comprehensive training dataset, but the potential increase in the NER system's performance may be of greater use than its ability to label uncommon or ambiguous instances of an entity type. It may also give greater certainty of how the NER will perform in certain use cases if it has a more focused and accurate domain to which it can be applied.

This is not an alternative solution to simply annotating more data, as that will both increase the number of annotated instances of an entity type and likely reduce the frequency of unique words somewhat without losing comprehensiveness in the annotated dataset. However minimising the complexity of the annotation scheme and the guidelines for annotation for a given task could alleviate the annotation work required for an adequate NER for that task.

With regard to the current effectiveness of the NER and how useful it may be, a high disparity in the NERs performance can be seen between entity types. In its current state, the NER most reliably labels User instances, with instances of Data and Tech also being identified with a reasonable accuracy. For these entities it has the potential to be useful in applications where high recall is not critical, such as a chatbot recommending data governance actions in response to messages, as even if some entity occurrences are missed, the bot can still be of benefit to any users, and help built a more complete data governance history for organisation.

While the performance requirements of an NER vary significantly depending on the task it is being used for, the performance of the current NER on the other 5 entity

types is likely too low to be useful in most applications, as the low precision and very low recall would mean very few entities would be correctly identified.

This can be addressed by increasing the size of the training set, which would likely improve the NER's performance sufficiently to make it useful in detecting some of the intermediately performing entities such as *Dmtask* and *Issue*, as well as further increasing its performance on the best performing entities. However the worst performing entities, particularly *Role*, will likely never be accurately predicted by the NER, even with a larger training dataset, and should either be removed from the annotation scheme, or have the annotation guidelines revised to reduce the diversity of words in the resultant dataset.

6. Conclusions

In this chapter the results of the implementation are reflected upon with respect to the goals initially set out in this dissertation. A summary of potential future work is also given.

In [section 1.2](#), this dissertation set out the research question '*To what extent can CRF-based Named Entity Recognition be used to extract data governance information from an enterprise chat channel?*' Three research objectives were then set, aiming to answer this question.

Research Objective 1: Survey the state-of-the-art to select the most appropriate methods and tools to be used in this work.

In [section 3](#) a variety of state-of-the-art works were explored which influenced this dissertation, from the only known application of natural language processing in data governance, to a range of state-of-the-art NER systems used in related applications. The tools and technologies used in the implementation of this work were also surveyed and selected from a range of options.

Research Objective 2: Create a corpus of enterprise chat data with data governance entities annotated upon which to train a Named Entity Recogniser.

A corpus of annotated enterprise chat data was created using an annotation scheme which defined 8 data governance related entity types and 3 relationship types. The resultant corpus contains ~5000 annotated words. This process is detailed in [section 4.1](#).

Research Objective 3: Evaluate and optimise the labelling accuracy of a general Named Entity Recogniser trained on this data.

With Stanford NER chosen as the NER system for this work, scripts were created to evaluate the NER system and gather a range of results from this evaluation. Using this the hyperparameters of the NER were varied to find the best configuration. This process is detailed in [section 4.2](#) and the evaluation covered in [section 5](#).

With regard to the research question itself, the evaluation carried out in this dissertation has explored the effectiveness of an optimised general named entity recognition system in identifying a set of data governance related entities from an enterprise chat channel. While the accuracy of the NER system varied heavily between entity types, the results obtained in the evaluation are very promising with the NER system showing very good results for 3 of the 8 entity types. As an initial system trained on a small dataset, this indicates the potential for future NER systems trained on a larger dataset to achieve a very high accuracy in extracting data governance information. The precise extent to which the current NER system can extract data governance information is detailed in [section 5](#).

6.1. Future Work

This dissertation has achieved the objectives that were set out and given an initial answer to its research question. However there are many ways in which this work can be expanded in the future. This section will highlight some of these potential means of continuing this work.

Annotate a larger dataset: The most straightforward way to improve upon the NER systems performance in this work would be to annotate a larger corpus of enterprise data governance chat. This should be a simple but time consuming task that would lead to reliable increases in the NER systems performance, particularly in recall.

Revise the annotation scheme: The annotation scheme used in creating the dataset for this dissertation was created to capture key data governance related entities. In [section 5.5](#) the factors affecting the NER system's effectiveness on each entity type are explored, and [section 5.6](#) proposes some possible ways in which the annotation scheme could be revised to improve the NER systems performance.

Investigate the use of alternative feature extractors: In this dissertation only the range of general feature extractors supplied by Stanford NER were explored. An investigation into the development of new feature extractors specifically aimed at extracting word features related to data governance entities could lead to improvements in the NER system's performance. Neural network based feature extractors could be of particular benefit to the NER system's performance, with neural network based feature extractors are used in the current best performing general purpose NER systems.

7. Appendices

7.1. Data Annotation Notes

7.1.1. General ideas

1. Try to limit annotations to the minimum number of words under one tag that gets across the concept, ideally one word.
2. I often refer to the Proposed Slack Channel Entities for Extraction doc to remind me of entity definitions and the examples and links to Collibra docs really help as there are some (IMO) non-intuitive parts to the model like metrics being business rather than governance assets.
3. When a multi-word phrase identifies a concept but contains other concepts too, eg “data backups”, annotate each token separately and link them by a REL relation ie “data^DATA backups^DMTASK, DATA-REL->DMTASK”. This gives us a weaker version of multi-level annotation and allows us to tag lots of entities that might be missed and gives examples of common patterns like <data owner> <data> as labels for datasets that I think are important to learn.
4. In Collibra DGOM the Business Asset type is potentially very expansive as it includes the terminology of the domain of discourse of the business (business terms). In many ways my v2 annotation model is an attempt to tag things that otherwise would have ended up in BUS. To a certain extent this means BUS remains a default tag.
5. Note that technically I have broken DGOM by using relation annotations between DMTasks (or roles, users) and assets since technically I think DGOM defines REL as between Assets only. However IMO it is too useful to avoid doing it.
6. I have tended to annotate in the spirit of helping the NER identify phrases that are likely to be useful for data governance even when the Slack is really talking about software development issues rather than data governance ones eg I may annotate “they really want feature X” as an issue, despite it not being a data governance issue (its a software development issue) but I think that form of words could be used to identify data gov issues. Feedback from Alfredo on this appreciated.
7. Users. When annotating these I think a good principle is to just tag the Slack user-names rather than any generic person name. We have no assurance

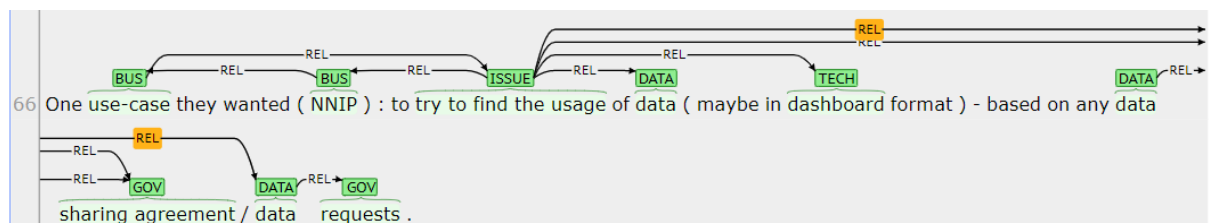
that they are CDG users as they could be customers etc. That is my guideline.

8. Web-pages/URLs. Most often these are TECH as they are the system. However sometimes they are data as it is the info at the URL that is being referred to rather than the web server/file/DB. More rarely they are an abstract concept like an ISSUE since people use the URL to refer to the concept they are discussing rather than the system or data.
9. Data vs Tech vs abstract governance concept. Often you have to decide whether to annotate a token as one of these. Are they talking about the system where the symbol is stored, the symbol definition or the concept the symbol stands for. Eg a bug report DB, the data describing a specific bug report, the bug report as a governance issue.
10. Role tends to be the rarest tag in my experience, recall that it is only for data governance roles not generic business roles.
11. Relations. When in doubt just use REL. I would like to have a tag for "Same As" so we could link terms referring to the same concept in a sentence, but maybe that is not needed by the NLP tools as they can detect that. Uses is the next most common and GovBy is the least frequent. Unlike other relation types it can only be between a governance asset and another asset. All others can be from any asset to any other asset type. Remember that all relations are bi-directional (symmetric?), even for ones like "Governed By" that sound like they should have an inverse. This means that you don't have to worry about the direction when annotating.

NB Now that I write this down I realise it would be better to be careful from the direction as then we can infer the inverse. However I have been sloppy since I started with REL which is symmetric and the tool UI is fiddly for relations so it discouraged me from being careful.

7.1.2. Example Annotation Process (Think out loud)

Line 66, chat_data/v2/tok0_3



I scan the line looking for easy/obvious entities. I hit use-case and decided it was relevant but hard so went on. NNIP is obviously the name of a company so I tagged that as BUS (part of the business's glossary/terminology). Then I found data which I usually tag as DATA, then I saw dashboard format which I was not sure whether they were talking about the dashboard or some data format that feeds it so I skipped. Then I saw data sharing agreement. This looks to me to be a data governance policy. I am always looking for these are IMO they are often subtle and I jumped on this one tagging all 3 words as GOV. Then I reconsidered and tagged data as DATA and sharing agreement as GOV with a REL between them. Then I looked at data requests and considered if that is really a metric they are measuring (BUS)? I decided this was hard and went back to look at the logic of the sentence.

Now I knew what the sentence is about I decided that use-case must be a BUS (business term). I saw "to try and find the usage of data" as the essence of the request so I tagged that as an issue. I skipped "to" since I wanted to minimise the words I tagged (although issues tend to be more words on average. Now is this really, really a data governance issue? Overall is a product management issue about how to satisfy a customer's desire for implementing a data governance issue so I was happy enough. I considered tagging usage as a metric but discarded it. Maybe I should have. However once I had decided what this issue was about I could see that the dashboard was what was important rather than the "dashboard format" as I am guessing that format here means "form-factor or use case" rather than data format. Hence I tagged dashboard as tech. Then I also thought that "data requests" is actually a form governance policy or request process so I tagged it as that...but I'm still not sure, maybe it would be better as BUS. I tagged data separately (as data) and linked it to requests with a rel.

Then I went to work on linking together the entities in the sentence, mainly linking the core issue to the things that it depends on/refers to. When doing this I try to avoid multiply linking all concepts that are already connected so issue links to the sharing agreement but not to the data linked to that sharing agreement. Now that I look at it again I can see that I am inconsistent since I linked issue to sharing agreement (implicit link to the data associated with that sharing agreement) but I also linked issue to data (implicit link to requests).

Sometimes when I feel that there is a genuine, independent link between the interlinked entities I will include additional paths so issue is linked directly to be NNIP

and use-case despite there being a implicit connection once I have linked to one of them since there is a relation between NNIP and use-case. I find it hard to give a rule about this but it is I suppose related to whether the entities represent different concepts or compound concepts (NNIP + use-case vs data + usage).

8. Bibliography

- Atdağ, S. & Labatut, V., 2013. *A comparison of named entity recognition tools applied to biographical texts*, IEEE.
- Baldwin, T. et al., 2015. *Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition*,
- Cherry, C. & Guo, H., 2015. The Unreasonable Effectiveness of Word Representations for Twitter Named Entity Recognition. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Available at: <http://dx.doi.org/10.3115/v1/n15-1075>.
- Derczynski, L. et al., 2015. Analysis of named entity recognition and linking for tweets. *Information processing & management*, 51(2), pp.32–49.
- Finkel, J.R., Grenager, T. & Manning, C., 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pp. 363–370.
- Forman, G. & Scholz, M., 2010. Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement. *ACM SIGKDD Explorations Newsletter*, 12(1), pp.49–57.
- Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio, Investigation of Recurrent-Neural-Network Architectures and Learning Methods for Spoken Language Understanding. Available at: <http://www.iro.umontreal.ca/~lisa/pointeurs/RNNSpokenLanguage2013.pdf>.
- Hripcsak, G. & Rothschild, A.S., 2005. Agreement, the f-measure, and reliability in information retrieval. *Journal of the American Medical Informatics Association: JAMIA*, 12(3), pp.296–298.
- Jannet, M.B. et al., 2014. *Eter: a new metric for the evaluation of hierarchical named*

entity recognition,

Khatri, V. & Brown, C.V., 2010. Designing data governance. *Communications of the ACM*, 53(1), pp.148–152.

Klein, D. et al., 2003. *Named entity recognition with character-level models*, Association for Computational Linguistics.

Makhoul, J. et al., 1999. *Performance measures for information extraction*, Herndon, VA.

Manning, C.D. et al., 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*. pp. 55–60.

Powers, D.M., 2011. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.

Richards, B., 1987. Type/token ratios: What do they really tell us? *Journal of child language*, 14(2), pp.201–209.

Shah, J., 2017. Utilizing Natural Language Processing and Artificial Intelligence to Identify Plausible Data Requests on Slack and Linking it to Collibra's System of Record Tool DGC. , p.17.

Smith, N.A., 2011. Linguistic Structure Prediction. *Synthesis Lectures on Human Language Technologies*, 4(2), pp.1–274.

Stenetorp, P. et al., 2012. BRAT: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 102–107.

Tong, F., Luo, Z. & Zhao, D., 2017. A deep network based integrated model for disease named entity recognition. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. Available at: <http://dx.doi.org/10.1109/bibm.2017.8217723>.

Wang, H. et al., 2007. A conditional random fields approach to biomedical named entity recognition. *Journal of Electronics* , 24(6), pp.838–844.