EXPLAINABLE ARTIFICIAL INTELLIGENCE: INTERPRETING NAMED ENTITY RECOGNITION USING POST-HOC TECHNIQUES

Sayan Banerjee
Student ID: 922994

A thesis submitted in fulfilment of the
requirements for the award of the degree of
MASTER OF SCIENCE IN MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

LIVERPOOL JOHN MOORES UNIVERSITY (LJMU)

AUGUST 2020

ACKNOWLEDGMENT

I would like to thank my thesis supervisor, Mr Suvajit Mukhopadhyay, for his valuable and timely review and feedback for the completion of the thesis. I would also like to thank DR. Manoj Jayabalan from Liverpool John Moores University for his continued support and guidance through weekly and one-on-one sessions throughout the duration of the program.
I am grateful to my parents, my wife, and my daughter for supporting and encouraging me throughout the duration of my study and writing this thesis.

Thank you,
Sayan Banerjee

ABSTRACT

Named Entity Recognition (NER) is a common Natural Language Processing task for extracting important information from documents. This technique is used very frequently in any organisational setup where the requirement is to extract key information from the digitised documents so that the information can be fed to further downstream applications. Organisations expect both accuracy and trust in the information extraction before they can use this process in practice. In recent days, Deep Neural Network (DNN) based architectures can achieve state-of-the-art accuracy in NER tasks. However, high accuracy alone is not enough to build trust amongst users, especially when black-box models like DNN is used. With the increasing use of deep learning in fields like image, text, etc., various model specific and model agnostic ways of AI explainability techniques have been proposed in recent years. Literature review suggests insignificant exploration in NER model explanation; only a couple of studies had experimented in this area. Only one of these studies used Local Interpretable Model-agnostic Explanations (LIME) to explain NER models. However, both studies lacked in demonstrating the evaluation of explanation with respect to ease of human consumption. This study demonstrated how a more recent model agnostic technique named Kernel SHAP, proposed based on LIME and classical Shapely Values from game theory, have been used to generate explanation for a DNN based NER model. The study experimented with several qualitative evaluation metrics to compare explanations generated from LIME and Kernel SHAP to understand whether they answer the questions that human expects from an explanation. The study showed how insights gathered from explanations generated by Kernel SHAP is used to improve the NER model.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

ABBREVIATIONS

Table 1: List of abbreviations

Abbreviation	Expansion
NER	Named entity recognition
NLP	Natural Language Processing
IE	Information Extraction
CRF	Conditional Random Field
ML	Machine Learning
Bi-LSTM	Bi-directional Long Short-Term Memory
XAI	Explainable Artificial Intelligence
DNN	Deep Neural Networks
LRP	Layer-wise Relevance Propagation
SA	Sensitivity Analysis
RNN	Recurrent Neural Network
t-SNE	T-distributed Stochastic Neighbour Embedding
LIME	Local Interpretable Model-Agnostic Explanations
Deep LIFT	Deep Learning Important FeaTures
SHAP	SHapley Additive exPlanations
POS	Parts of Speech
GloVe	Global Vectors for Word Representation
AOPC	Area over the perturbation curve
NLTK	Natural Language Toolkit
HCI	Human-Computer Interaction
MT	Machine Translation
VAE	Variational autoencoder
TDS	Text Deconvolution Saliency
t-SNE	T-distributed Stochastic Neighbour Embedding
AIC	Akaike Information Criterion
BIC	Bayesian Information Criteria

CHAPTER 1: INTRODUCTION

1.1 Background of the Study

In the domain of Artificial Intelligence (AI), Natural Language Processing (NLP) is the field that focuses on how computers can process and interpret natural language data. Due to the complexity of any natural language, the study of natural language is subdivided into various sub-fields (Bird et al., 2009).

Named entity recognition (NER) is a classic NLP task which comes under Information Extraction (IE) sub-field. This is a very common but non-trivial task in retrieving information from documents. NER is a sequence prediction problem where each word in a sentence (i.e. a sequence of words) are marked whether that is part of an entity or otherwise. NER modelling is challenging as entity prediction for a word in a sentence is not only dependent on that word, it is also dependent on the contextual words. What makes this task more challenging is that only a small subset of words in a sentence is part of an entity. This makes evaluation of a NER task tricky as accuracy of NER prediction depends on accuracy of the predicted entity tag for each word in the sentence. If there are multiple words in an entity, inaccurate prediction for one word in the entity makes the prediction wrong for that entity.

One of the widely used probabilistic models for NER has been Conditional Random Field (CRF) as proposed by (Lafferty et al., 2001). It is possible to interpret a CRF model by looking at the feature weights calculated by the CRF algorithm. Additionally, the features are carefully curated by Machine Learning (ML) practitioners for CRF, therefore interpreting the features does not need additional effort. This is almost the same as logistic regression where coefficients of features represent their importance. Training a CRF means to compute the optimal weights for each feature given the observed sequence of words.

Recently, Bi-directional Long Short-Term Memory and Conditional Random Field (BI-LSTM-CRF) architecture, as proposed by (Huang et al., 2015), has become very popular for sequence tagging task. This architecture and its variants have been used in multiple recent studies (Lee, 2017; Luo et al., 2018; Wei et al., 2019). This has proved to deliver better performance in entity extraction from text. However, any deep neural network model achieves its performance because of its complex architecture and ability to efficiently solve

the huge parameter space and this makes deep learning algorithms black-box (not transparent by design) model (Arrieta et al., 2019). Therefore, the NER model created using BI-LSTM-CRF does not offer the same interpretability compared to a NER model created using CRF alone.

These issues can pose serious problems in utilizing the NER output from BI-LSTM-CRF based model for any practical use. For instance, in an organizational setup, where organizations have migrated to digital document repository long back, it is often seen that extracting important information from the digitised documents is not an easy task as documents have not been properly tagged or indexed to start with. When these organizations plan to use the information available in the documents for any further processing, Machine Learning practitioner resort to NER to extract important information. Both poor performance and lack of interpretability in entity extraction become a hindrance in using the predictions for any downstream application as errors in information extraction potentially would lead to undesired consequences.

For example, a common practice in the industry is to use a small set of skilled people to do the annotations (tagging the words with entity names) for NER training and validation data sets. This has a higher probability of annotations getting impacted by the human bias of the annotators. Usual training and validation step in the model building phase will not help to identify any unreasonable bias introduced in annotation and therefore the NER model will learn them as well. A simple example of such bias would be tagging unfamiliar proper noun as part of an address if the previous word is a number. This bias will result in tagging all the words in "1. Mao Ze Dong, 2. John Q. Public, 3. Aditya Pratap Singh Chauhan" as part of address where it is a numbered list of persons' names[1]. If an explanation of the NER model can highlight this problem in annotations then it will be possible to review these annotations and improve the model.

An extensive survey on eXplainable Artificial Intelligence (XAI) (Arrieta et al., 2019) stresses upon the point that performance metrics/accuracy of an ML model may be of interest for research studies, but the understanding of the ML model is necessary for

[1] The names are for illustrative purpose only, quoted from https://www.w3.org/International/questions/qa-personal-names

practical use and further enhancement. The survey states the following regarding consideration of interpretability while developing an ML model:

- Interpretability of an ML model makes it easier for ML practitioners to make the model accurate by removing unnecessary biases induced by training data.
- Interpretability in a model increases the trustworthiness of the ML model and therefore increases chances of adoption of the ML model in any business process.

There are various types of explainability methods depending on whether the ML algorithm itself is interpretable or need a post-hoc explanation, how the results of explanations are presented to the user, whether the explainability method is specific to a class of model or can be applied on any class models, whether they explain the complete model or specific set of predictions (Molnar, 2019). Explainability techniques for black-box models are referred to as post-hoc explainability methods. The primary objective for post-hoc explainability method is to make the output of a non-interpretable model meaningful to a human being. Please note that DNNs are not only non-interpretable models; SVM and Tree-based ensemble models are also considered to be back-box models as they do not provide transparency by design.

The survey performed by (Arrieta et al., 2019) suggests various available approaches for post-hoc explainability. Figure 1 depicts all different post-hoc explainability approaches (Arrieta et al., 2019).

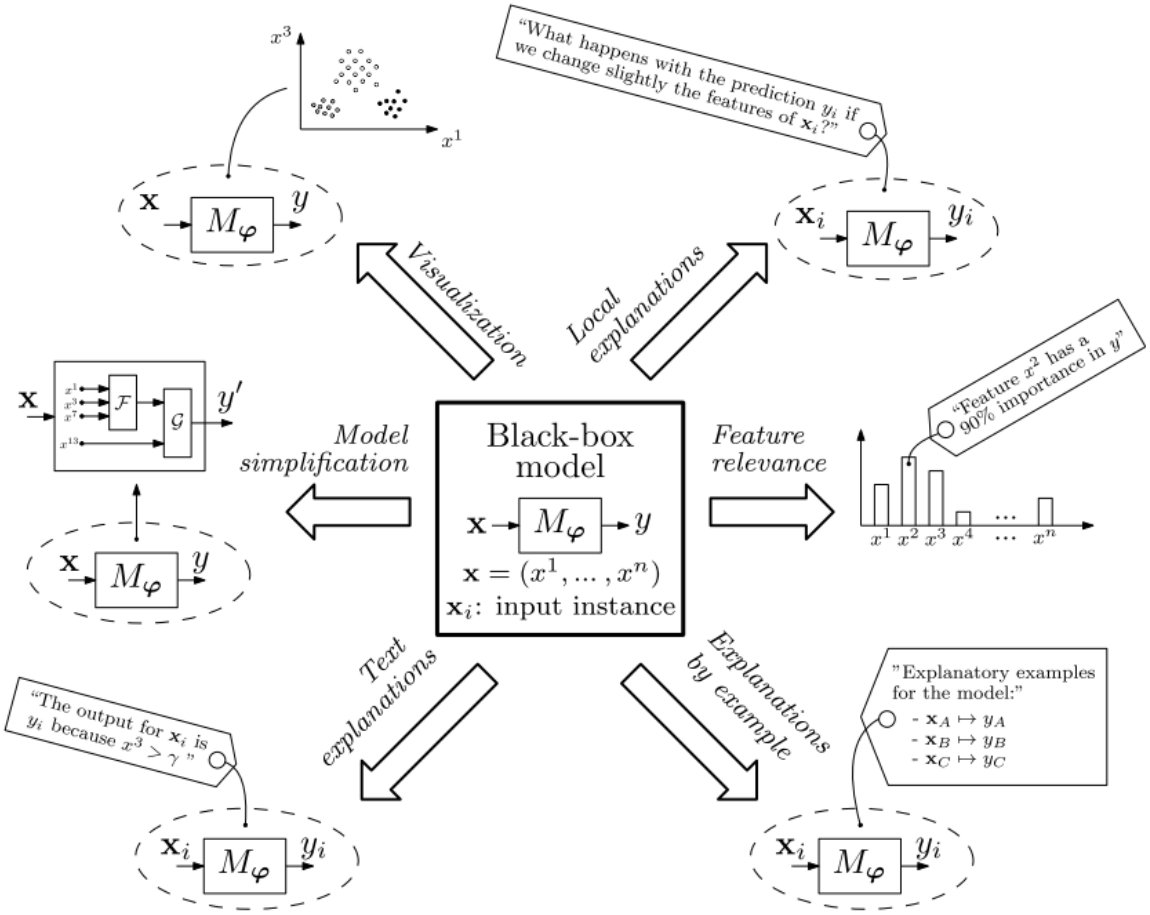

Figure 1: Post-hoc explainability approaches (Arrieta et al., 2019)

In the review of multiple model specific and model agnostic post-hoc explainability methods (Arrieta et al., 2019), it is observed that explainability techniques for DNN that got evolved in recent years are of two types primarily - feature relevance as explanation and local interpretability as explanation. Feature relevance explainability techniques are suitable when global interpretability of an ML model is desired. This technique calculates the importance score for each feature at the input with respect to output. Feature relevance techniques can be both model-agnostic and model-specific. It is also noticed that recent research has more coverage on local explainability methods as it is difficult to generate truly global explanations that can be consumed by human users. Local explanation of a model refers to the interpretation of predictions on a single or set of observations. These methods, in general, try to approximate the model using an interpretable function locally to the set of observations. Majority of these techniques are model agnostic.

Review of AI explainability for NLP highlights that expectation form explanation of NLP task is more than the explanation of other types of ML models (e.g. a DNN model on image). Introduction of DNN for NLP tasks replaced rich human interpretable feature engineering steps with end to end DNN systems. Therefore, the role for explanation in NLP task is also to demonstrate how these feature engineering steps are captured inside the DNN model (Belinkov and Glass, 2019). In the literature review, it is noticed that AI explainability techniques were not studied on deep learning based NER models extensively. This provides an opportunity for the current research to study recently developed AI explainability techniques on deep learning based NER models. This study will focus on a couple of model-agnostic local explainability techniques that are relevant to the current study on interpretability of NER.

While there has been significant research on AI explainability methods, multiple studies (Doshi-Velez and Kim, 2017; Carvalho et al., 2019; Molnar, 2019) agreed that there is lack of defined metrics for evaluating explanations generated from AI explainability methods. The study will explore several qualitative and quantitative metrics for evaluating explanations. This will help to assess whether explanation generated for NER models can build trust for the predictions and whether the insights from explanation are useful in improving model performance.

1.2 Problem Statement

In the recent past, with the increasing use of various Deep Neural Network based architectures for solving many complex problems, the need for explainability has increased multi-fold. There could be various types of stakeholders for these explanations, an ML practitioner might need an explanation of model to make it better by removing undesired biases, an expert user might look for explanations so that the system can be trusted while in use (Ribera and Lapedriza, 2019).

Review of AI explainability for NLP tasks suggests that there is a lack of study in locally interpretable explanation (Volpato, 2019). It is noticed that the work in AI explainability for NLP had primary focus on classification. There exists very little work where sequence to sequence tasks are the primary subject of explanation. In the review of AI explainability for NER tasks, two studies are found that experimented explanation on NER tasks. (Xie et

al., 2018) demonstrated the use of Layer-wise Relevance Propagation (LRP) to interpret the Bi-LSTM-CRF model used in Named Entity Recognition. Primary disadvantage for using LRP was that the explanations were not suitable for layman users and CRF layer could not be explained using LRP being DNN specific explanation technique. The second work on NER explainability was performed by (Villarroya, 2018). The study used an improvised version of LIME - Local Interpretable Model-Agnostic Explanations (Ribeiro et al., 2016) that can provide locally interpretable explanations for entities instead of individual words (entities can be multi-word). LIME generates explanation by building locally interpretable model using a new dataset created by perturbing the original data set. The process for perturbing the data set is different for tabular data and text. (Villarroya, 2018) adjusted the algorithm to ensure entity level explanations. The primary disadvantage of using LIME is that assigning weightage for perturbed observation while building the locally interpretable model needs to be done heuristically. There is no proven method to arrive at optimal weights. If the weights are not defined right, then the locally interpretable model created from perturbed data will not be faithful. To overcome this difficulty, this study proposes the use of Kernel SHAP for explaining the NER task. Kernel SHAP is proposed in the paper that introduces SHapley Additive exPlanations - SHAP (Lundberg and Lee, 2017). This method is based on both LIME and Shapley values. Shapley values is a cooperative game theory approach for fair distribution of outcome to its players depending on the level of contribution. The advantage of using Kernel SHAP over LIME is that it removes the need for heuristically choosing the weights for observations in perturbed data set as it is decided by the Shapley Values.

It is noticed that current explainability techniques lack focus on how explanations could be made more useful for human use. (Miller, 2019) argued that the explanation should have properties that can answer "*why*" question with reference to another fact (i.e. contrastive) and should focus of few selective explanations (especially if there are unusual causes) that serve the purpose of the users. Studies, (Ruping, 2006; Guidotti et al., 2018; Carvalho et al., 2019), discussed qualitative properties of explanation, like accuracy, comprehensibility, and time efficiency, that is required for human users to consume the explanation. There exist studies that define qualitative evaluation metrics based on type, volume, and structure of explanation (Doshi-Velez and Kim, 2018; Carvalho et al., 2019). However, there is a lack of adoption of these metrics for evaluation. Therefore, this study plans to focus on the

evaluation of explanation using the set qualitative metrics that can assess human acceptability of a generated explanation.

A potential concept for quantitative metrics is discussed which suggests that a similar set of observations should always demonstrate similar explanation and dissimilar set of observations should always demonstrate different explanations (Sundararajan et al., 2017). Though quantitative metrics is preferred, there is no defined set of quantitative metrics available for explanations that were used across multiple studies. Another set of studies, (Alvarez-Melis and Jaakkola, 2017; Gilpin et al., 2019), mentions that explanations should provide insights on the shortcomings of the explained model. Inspired by this thought, this study proposes evaluation of explanation based on its ability to guide model improvement.

1.3 Aim and Objectives

The main aim for this study is to generate explanations for Named Entity Recognition (NER) predictions from state-of-the-art NER model (built using BI-LSTM-CRF architecture) using Kernel SHAP which is a variant of SHAP (SHapley Additive exPlanations) framework. The primary goal of this research is to generate explanations for NER predictions that are easy to understand for layman users (i.e. having sufficient language and domain understanding but no formal ML knowledge).

The research objectives are formulated based on the aim of this study. Those are as follows.

- To generate explanations for NER prediction using Kernel SHAP that is proposed based on both Shapley values, a cooperative game theory approach for fair distribution of outcome to its players depending on the level of contribution, and linear LIME.
- To evaluate explanation generated from Kernel SHAP based on the attributes of a good explanation for human being considering the audience of explanation. Ability to answer contrastive "*why*" questions, select the most relevant explanations amongst all possible explanations and avoid using conflicting explanations are considered important constituents of a good explanation.
- To validate whether insights on shortcomings of the NER model (e.g. bias) gathered from explanation generated by Kernel SHAP is useful for improvement of the model.

1.4 Research Questions

Literature review leads to the following research questions for the current study.

- *Is Kernel SHAP an effective method to explain Named Entity Recognition model built using BI-LSTM-CRF architecture?* –this study would like to generate explanation for NER task using Kernel SHAP as proposed in (Lundberg and Lee, 2017) and compare the explanations generated using LIME-NER as suggested in (Villarroya, 2018).
- *Are the explanations generated using Kernel SHAP for Named Entity Recognition model easy to understand for layman users having sufficient language and domain understanding but no formal ML knowledge?* – The study by (Xie et al., 2018) is insufficient for generating explanation for NER model for users having language and domain understanding but no formal ML knowledge. This study would like to evaluate explanation based on the qualitative attributes that are necessary for easy human understanding of explanations.
- *Can the explanations generated using Kernel SHAP for Named Entity Recognition model provide insights related to shortcomings of the model?* - as suggested in the future work by (Xie et al., 2018), this study would like to investigate how NER explainability generated using Kernel SHAP can be used to make improvements in the NER model.

1.5 Scope of the Study

The scope of the study will be limited to the following items

1. NER model building on a subset of CoNLL 2003 English named entity dataset (Language-Independent Named Entity Recognition (II), 2020) and validation of the NER model using sequence F1 score
2. Generating explanation using LIME-NER as implemented in (Villarroya, 2018) and visualization of explanations
3. Generating explanation using Kernel SHAP as proposed in (Lundberg and Lee, 2017) and visualization of explanations
4. Comparison of explanations generated from both the methods using qualitative and quantitative evaluation techniques.

5. Building a revised NER model using insights gathered from explanations and compare the performance of the revised model against the initial NER model.

The following items will not be in scope for the study.

1. Generating explanation using another SHAP method - Deep SHAP (DeepLIFT + Shapley values), even this technique might be suitable due to the DNN architecture of the NER model.
2. Non-English language for NER model building.

1.6 Significance of the Study

The study will have the following contribution to the body of research.

- The study will explore how AI explainability can be used to generate explanations for NER task using one of the most recent techniques in the AI explainability space. Kernel SHAP, as proposed by (Lundberg and Lee, 2017), have not been used to explain NER predictions so far. Therefore, this study intends to explore this new area in AI explainability domain.
- The study aims to provide a comparison between explanations generated using LIME (Ribeiro et al., 2016) and Kernel SHAP (Lundberg and Lee, 2017) in following 2 dimensions - (1) Interpretability of explanations generated for layman users having no / limited ML understanding. The comparison of LIME and Kernel SHAP explanations in NER prediction will contribute to the knowledge base in AI explainability domain. (2) Applicability of insights from explanations to improve NER model prediction performance. Comparison of LIME and Kernel SHAP in this dimension will enrich the knowledge around error analysis of NER model for ML practitioner community.

1.7 Structure of the Study

This chapter, chapter 1, introduces the problem domain of this study. It provides the necessary background on NER and AI explainability that is required for an in-depth understanding of the study. This section explains the problem statement that this study is focusing on and the aim and objectives of the study that will address the problem statement. It also discusses the research questions that the study will try to answer and its desired contribution in the space of DNN based NER modelling and AI explainability.

Remaining of this thesis is structured into five chapters. The subsequent chapter, chapter 2, reviews the literature of AI explainability extensively to understand the motivation of the same for ML models. That section has tried to compare a few recent post-hoc explainability techniques in the view of the aim and objectives of this study. This chapter has discussed present work on AI explainability for NER, the gaps in those works and ideas on improving them.

Chapter 3 discusses the proposed methodology and data in details. This section discusses the data format and required data pre-processing and transformations for NER model building using BI-LSTM-CRF architecture. It discusses the evaluation metrics for the NER model building. Later the focus of the chapter is on explanation generation for the NER model using LIME-NER and Kernel SHAP and required adaptations for the same. The study discusses the evaluation strategy for the explanations generated and how to present a comparison between explanations from LIME-NER and Kernel SHAP. Finally, this chapter discusses the novelty in methodology that will allow the use of Kernel SHAP for NER explanation and qualitative evaluation of explanations to assess its suitability to layman users.

Chapter 4 is broadly divided into four sub-sections. First sub-section discusses the design for data pre-processing and data transformation as per the methodology discussed in chapter 3. This section discusses each step of data processing and transformation in details. Then the second sub-section presents the BI-LSTM-CRF architecture for building the NER model and the hyperparameters used. The second sub-section ends with model evaluation steps. In the third subsection, the steps are detailed for post-processing NER prediction output to be used for generating explanations. The experiments for explanation generation using LIME and Kernel SHAP is also designed in this section. The final subsection provides design and implementation details for building qualitative and quantitative evaluation methods for comparing LIME-NER and Kernel SHAP explanations. This subsection also discusses the modifications on NER models that are inspired by the insights from generated explanations. Additionally, this section talks about the hardware and software resources used for the study.

Chapter 5 focuses on the results of the experiments implemented as par chapter 4. This section starts with the performance metric for the NER model. Then this section discusses explanations generated from LIME-NER and Kernel SHAP for various accurate and inaccurate prediction scenarios. These explanations are then evaluated based on three qualitative evaluation method and one quantitative evaluation method which helps in a comparative analysis of LIME-NER and Kernel SHAP explanations. This section ends with a discussion on the performance of modified NER models that are designed based on the insights from explanations.

Chapter 6 concludes this study by discussing how the study meets the aim and objectives set for this study. This section also discusses the research questions and how comprehensively the study could answer them. The contribution of this study in the space of AI explanation is discussed in the following subsection. Finally, this section discusses the limitations of this study and future recommendation from this study that can be used to extend research in the domain of AI explainability and NER.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This section will focus on previous work on AI explainability. This section will review how AI explainability became essential for current day AI, what are the different type of AI explainability techniques available and how they compare against each other. Noticeably, most of the explanation techniques were initially applied to image related tasks, potentially due to easy visualization of input feature space; and later adapted for text. Therefore, this study will first discuss the work where any new explainability technique got proposed and then how that is adapted for text related task. Eventually, the study will discuss the techniques used for explainability on DNN based NER models.

2.2 AI Explainability

2.2.1 What is AI explainability?

According to (Carvalho et al., 2019) and (Abdul et al., 2018), historically the interest in AI explainability started with experts' systems (Swartout, 1983) and production rules (Davis et al., 1977) as explanation in late 1970 and early 1980; later it got some momentum in the 1990s when Rule Extraction was used to generate interpretation for Artificial Neural Networks (Andrews et al., 1995) and Support Vector Machine (Núñez et al., 2002) models. In the similar period, researchers tried explanation in Bayesian Belief Networks (Madigan et al., 1997). Again in 2000, researchers introduced an explanation for recommender systems (Herlocker et al., 2000; Cramer et al., 2008). However, due to the shift in focus for AI research towards complex algorithm implementation to achieve higher prediction performance, research on AI explainability lost its pace. In recent days, when the implementation of black-box / non-transparent algorithms to solve complex problem has become prevalent, the need for AI explainability is felt from multiple perspectives, e.g. regulatory, social and technology etc. Following are a few motivating examples highlighting the need for ML models to be able to explain how they have arrived at their result.

In the survey of explaining black-box models (Guidotti et al., 2018), author have quoted several instances where the internal bias of ML models had inadvertent social impacts. For example, propublica.org (Angwin and Larson, 2016) had reported that

a risk-scoring model for potential future crimes was biased towards assigning a high-risk score to a particular ethnicity. Another study by (Caliskan et al., 2016) argued that training ML models on data that already have human bias results in a biased outcome. The study concludes that an opinion/sentiment mining model trained on a text corpus with unpleasant terms more associated with a particular ethnicity had a higher probability of suffering from similar bias in its outcomes. In another instance, businessinsider.com (Letzter, 2016) reported, while Amazon.com was using ML model to select areas in the US for their same-day delivery program, the software excluded certain minority neighbourhood unintentionally.

In recent advancements of DNN, it has achieved unparallel performance in various tasks related to image, text etc. However, there are multiple studies performed to show that little perturbations (sometimes indistinguishable in the human eye) in input can lead a DNN model to provide wrong results. For example, in the image domain, (Szegedy et al., 2014) shows that undetectable changes in an image can alter a DNN's predicted category for the image. (Nguyen et al., 2014) shows that DNN can predict a category for an image with very high confidence score though the image is completely unidentifiable in human eyes. Similarly, for text, (Liang et al., 2017) has shown that little perturbations introduced in a text can lead a DNN text classifier to misclassify the text in the desired category.

The "right to explanation" clause in European Union General Data Protection Regulation - GDPR (General Data Protection Regulation - Wikipedia, 2020), XAI program in U.S. Department of Defence funded "Defence Advanced Research Projects Agency" (DARPA) (Explainable Artificial Intelligence, 2020), USACM statement on algorithmic transparency and accountability (Principles for Algorithmic Transparency and Accountability, n.d.), are few prominent examples where regulatory bodies expressed the need of AI explainability in the public interest (Abdul et al., 2018).

The aforementioned examples emphasise on the need of the mechanism that will help users to assess whether and when outcome from an ML model can be trusted

for any decision-making process. It is important to note that AI explainability is much needed when the result of an ML model is used for decision making.

According to (Arrieta et al., 2019), there is a lack of consensus amongst researchers in the AI explainability domain regarding the intent of explainability and interpretability of ML models. As per DARPA website (Explainable Artificial Intelligence, 2020), the goals for explainable AI is building explainable ML models without compromising on performance metrics for present-day state-of-the-art ML models so that users can understand and trust the result of the ML models. It also says that explainable AI should be able to help users to effectively manage the models and model outputs. I.e. explainable AI should not only provide information on why it succeeds but also when it would fail and not trustworthy.

(Miller, 2019) defines explainability as "*an explanation is the answer to a why-question*". This paper uses the concept from social sciences to discuss the traits AI explainability should cover to satisfy human need. It states that human being is more interested in why-questions that are contrastive (i.e. Why does the model behave in manner A, instead of manner B?).

(Honegger and Blanc, 2018) argues that the definition of AI explainability is contextual, i.e. it depends heavily on the target application domain of the ML model. This paper has differentiated between explanation and interpretability – explanation is the path to the final goal of interpretability. It states explanation as human comprehendible characteristics of the ML model that eventually helps to increase the interpretability of the model (Carvalho et al., 2019).

In the subsequent section, we shall delve into the details of various objectives of AI explainability as mentioned in multiple relevant literatures.

2.2.2 Why AI explainability?

Three recent review papers on AI explainability (Abdul et al., 2018; Arrieta et al., 2019; Carvalho et al., 2019) have advocated multiple goals for explanation of the

ML model. Following goals are common in all these three papers and seems relevant in the context of the current study.

- Fairness - From a social setting, the explanation of an ML model should help in analysing the bias in the model. This would help to ensure that the results of the model are ethically correct and fair to all stakeholders.
- Trust – This is a primary purpose of explanation, but at the same time it is very difficult to quantify trustworthiness of a model. An explanation should be able to instil trust in the users of the model regarding its stability. However, it is important to assess the trustworthiness of a model along with the other goals of explainable AI.
- Causality – ML models do not directly model causality, instead, they model the correlation between data points. AI explainability is expected to take a step ahead and highlight the causal relations amongst the correlations. According to (Miller, 2019) causality resonates well with how human perceive explanation.
- Confidence / Reliability / Stability – The terms used in different literature for this concept is different, however, the objective is same. AI explainability should be able to measure the stability of an ML model, i.e. how much model behave differently for a small change in input. A stable ML model will be reliable to its users and thus will instil confidence in the user of the model.
- Accessibility / Interactivity – (Arrieta et al., 2019) mentioned this concept as the third most considered goal as per their literature reviews. This goal of AI explainability focused on how explanation can facilitate the use of an ML model easy for a non-technical and non-expert user.

(Arrieta et al., 2019) mentions few more goals of explanations, e.g. transparency (linked to accountability as being enforced by a regulatory body like GDPR), privacy awareness (also mentioned in (Carvalho et al., 2019)). Though these concepts are not elaborated in this review but would be equally important in a related context.

2.2.3 Human explanation and AI explanation

Almost all the literature on AI explainability has covered how AI explainability should relate to human explanation. All these literatures have taken their inspiration

from a small number of papers that had reviewed AI explainability in the perspective of psychology, philosophy, and social sciences.

The book - Interpretable Machine Learning (Molnar, 2019), survey paper from (Carvalho et al., 2019), thesis paper from (Honegger and Blanc, 2018) etc. followed the definition of explanation laid out by (Miller, 2019). The paper argued that explainable AI should not be "*more AI*". This should be seen as human-agent collaboration and that can be defined as the intersection of AI, human-computer interaction (HCI) and social science as shown in Figure 2 Venn diagram from the same paper.

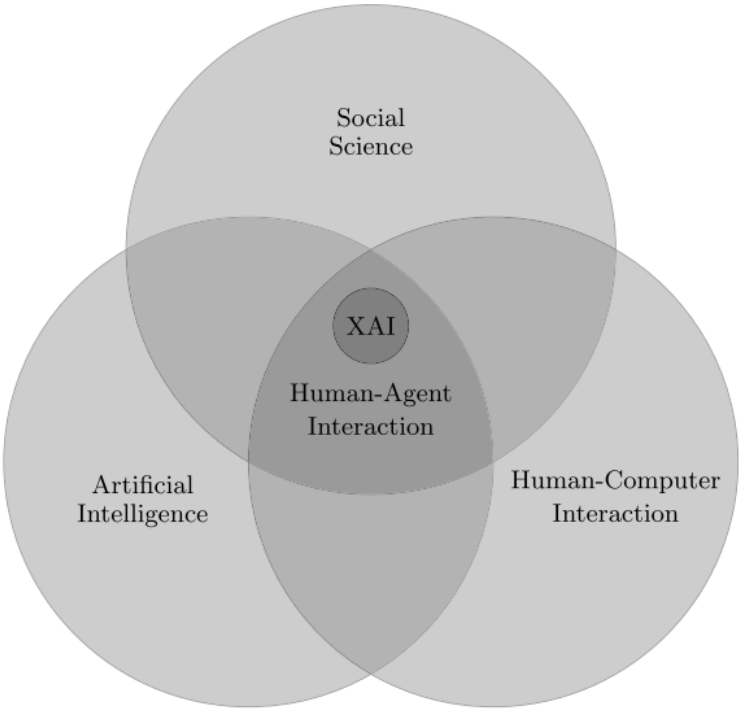

Figure 2: XAI as intersection of AI, human-computer interaction (HCI) and social science (Miller, 2019)

According to (Miller, 2019), an explanation should answer a "why" question. However, people do not ask why something happened, instead, people ask why event A happened instead of event B. Point to note here is that event B is generally influenced by the cognitive bias of the user. The paper summarised that explanations are not always presenting all causal relationships to the user; those must be contextual. In general, any event will have an infinite number of causes behind it, but users will be interested only a few selected causes based on the context and

cognitive bias. Based on several literature reviews on AI explainability, HCI and social sciences, (Miller, 2019) has discussed the following characteristics for good explanations. (Molnar, 2019) have summarised these characteristics in the book - Interpretable Machine Learning.

- *Contrastive explanations* – According to (Lipton, 1990), in general, human will have a pre-conceived notion for the outcome of any event due to cognitive bias. Therefore, a human will ask for an explanation of the outcome of an event is any different from what was perceived. This means, the user will not ask why an ML model is predicting something; instead the user will ask why the ML model has predicted A instead of B (which is influence by user's cognitive bias). Human is more comfortable to reason using counterfactual events. For example, in the present-day situation of COVID-19 pandemic, if an ML model built to predict mortality rate for a country/geography, age group etc., explanation of prediction will make more sense to a human when the prediction is explained in comparison to the mortality rate of another country/geography and/or another age group.
- *Selective explanation* – Human does not like to consume all the explanations possible for an event (in practice there could be a very large number of causes that are possible for a single event). It is likely that the user would always select only very few causes as a suitable explanation. The selection of causes as an explanation is contextual to the user. While building explainable AI, it is important to implement the selection of causes for an explanation based on the stakeholder of explanation (Ribera and Lapedriza, 2019). To continue with earlier COVID-19 pandemic example, selection of causes as explanations of mortality rate for a country/geography, age group etc. would differ vastly in number for layman users vs doctors treating patients vs research community trying to invent drug or vaccine.
- *Social explanations* – A good explanation from an AI system should be like social interaction between explanation generation method and the user of the explanation. The concept of stakeholder focused explanation form (Ribera and Lapedriza, 2019) is appliable here as well. As an extension to the example from the selective explanation, an explanation would differ vastly

in content and nature for layman users vs doctors treating patients vs research community trying to invent drug or vaccine.

- *Focus on unusual environment* – Nature of human being is to focus on abnormal causes while reasoning through an outcome of an event, even if the outcome is natural in a normal circumstance. This means the inclusion of an unusual scenario from input space makes the explanation more valued for AI explainability. For example, in COVID-19 pandemic scenario, it will be more meaningful to be able to explain high mortality rate for elderly age groups with the reason that older people are likely to be severely ill once infected and people from that same age group is likely to have pre-existing medical conditions (instead of associating the event only with the age group) (Myth busters, 2020).
- *Truthful explanations* **–** Ideally the explanations should be truthful to the event, however, this contradicts selective explanation clause. (Molnar, 2019) emphasised that selective explanation is more important than the truthfulness for a human being to understand the explanations. However, the truthfulness of explanation is supposed to vary depending on the stakeholder of the explanation. This means the ML model should be completely truthful to the event so that truthfulness in the explanation can be adjusted based on the audience of explanation.
- *Consistency with prior beliefs* – Human tends to accept only the explanations that confirm their prior beliefs. In psychology, this is known as confirmation bias (Nickerson, 1998). This sometimes contradicts truthful explanation. (Honegger and Blanc, 2018) pointed out that satisfying both truthfulness and confirmation bias while generating explanations may not be possible. In the ML model building, introducing features conforming to confirmation bias may be a way to test whether the confirmation bias holds while generating explanations.
- *General and probable causes as explanation* – In absence of unusual cause, a general cause that can explain many events are considered good explanation. Though unusual causes, if available, still take precedence in making an explanation better.

2.2.4 Type of explainability techniques

The book - Interpretable Machine Learning (Molnar, 2019) summarised the different criteria based on which machine learning interpretability can be categorised. (Carvalho et al., 2019) followed the same categorisation as well.

- Intrinsic vs post-hoc:
    - Intrinsic explainability refers to the ML algorithms that are transparent by design. The model building methodology ensures that the model outcome would be interpretable without any additional effort. This is achieved by restricting the complexity of the model. Linear and logistic regression, decision trees, decision rules, rule fit, K-nearest neighbour, Naïve Bayes, General additive models etc. are the algorithms that provide an intrinsic explanation.
    - Post-hoc explainability refers to generating an explanation after the model is trained. Feature importance is an example of post-hoc explainability method. In general, post-hoc explainability can be applied on intrinsic models as well provided the post-hoc method supports that transparent algorithm.
- Result of explanation: This categorization is based on how the result of the explanation is presented to the user.
    - Feature summary: Some explanation methods provide feature-wise summary using any statistical method. Often it is a single number against each feature. Most of the times the explanation is consumed through visualization. The output feature importance, permutation feature importance falls into this category; though this can be consumed as a table or through charts. The output of Partial dependency plot, Individual Conditional Expectation, Accumulated Local Effects also falls under this category but can be consumed meaningfully using visualization alone (Molnar, 2019).
    - Model internals: The techniques in this category uses inner details of the model to express explanations. As the output of these techniques is tightly coupled with model internals, these explanation methods are model specific. For example, using the coefficients in a linear model for explanation falls into this category (note this can be used

as a feature summary as well). In the world of DNN, one such example is feature visualization in CNN that helps in visualizing the input that maximises activation for a neuron, or a channel, or a layer or final prediction (Olah et al., 2017).

- Data points: This explanation method uses exiting or newly created data point for explanation of the ML model. For a data point to work as an explanation, the data point would need to be interpretable and therefore this class of explanation works better with images and text compared to tabular data. The book - Interpretable Machine Learning (Molnar, 2019) refers to these explanation techniques as "*Example based explanation*". Counterfactual explanation falls under this category (Wachter et al., 2017). A counterfactual explanation is a model agnostic explanation method that explains in terms of the smallest change required in an input feature to change the output prediction. Another example of data points based explanation is Anchors (Ribeiro et al., 2018) that highlights the features that are sufficient for a prediction.
- Local surrogate interpretable model: This is a two-step process. In the first step, the input-output of an ML model is approximated using one of the intrinsically interpretable models. Then feature summary and model internal wights of the surrogate model is used to explain the target model.

- Model-specific vs model-agnostic:
  - Model-specific explanations are specific to the algorithm. This category of explanation methods is tightly coupled with the internal working of the model. For instance, explanation generated based on backpropagation can only work on neural network architecture, e.g. Saliency maps (Simonyan et al., 2013). A more specific example is deconvolution network (Zeiler and Fergus, 2014) that can provide examples on Convolution Neural Network alone.
  - On the contrary, model agnostic explanation methods can be applied to any class of models (including intrinsically interpretable models). This type of explanation methods is not dependent on the inner

working of the model which essentially decouple the explanation from the model. This type of explanation methods uses input and output pair to arrive at explanations. E.g. LIME (Ribeiro et al., 2016)

- Local vs global:
    - Global explanation ideally refers to a holistic explanation of the model. (Carvalho et al., 2019) noted that it is very difficult to achieve a global explanation for any complex model as it needs an understanding of both features and algorithm. However, it is practically possible to achieve global explanation on a modular level, e.g. using coefficient values in linear regression.
    - The local explanation is focused on explaining a single or a very small set of predictions. The general idea of local explanation is based on approximation of prediction on a limited input space using intrinsically interpretable models. An example of a local explanation is LIME (Ribeiro et al., 2016)

The book - Interpretable Machine Learning (Molnar, 2019) suggests that the future AI explainability will be using model-agnostic explanation techniques as these methods can scale better and can be easily applied on any class of models. It also suggests that intrinsically interpretable models will not lose interest as that will be able to serve the purpose of explainability by design, but the use may be limited to simpler data sets.

2.2.5 Evaluation of explanation

Literature survey suggests that research on explainable AI is more focused on building new explainable techniques. AI explainability research on evaluation of explanation is very limited (Carvalho et al., 2019; Molnar, 2019).

Interpretable Machine Learning book (Molnar, 2019) covers the approaches for evaluation of explanation from (Doshi-Velez and Kim, 2017). (Carvalho et al., 2019) cited the same source on the formulation of explanation evaluation approach.

- *Application level evaluation (end task)* – Here an expert user would need to evaluate the explanation generated by a real-life system. As both application and expert users are in the loop for this evaluation, the quality of the

evaluation is directly proportional to the quality of application set up and ability of the expert user to explain an outcome. As application-level evaluation need expert user effort and real-life application setup, the cost for this type of evaluation is higher than the other two types of evaluations. However, this evaluation method is likely to have more validity compared to the other two methods.

- *Human level evaluation (simple task)* – This is a human-assisted evaluation as well; however, they would be layman users and not expert users. With respect to the cost and validity of this evaluation, this holds the middle ground. I.e. cost and validity of this method is lower than application-level evaluation but higher than function level evaluation.
- *Function level evaluation (proxy task)* – This does not need any human intervention. Instead, this uses already evaluated qualities of interpretability (potentially evaluated by a human-level evaluation) as a proxy task for evaluating explanation. Depth of decision tree, the sparsity of linear model are examples of function level evaluation. i.e. human-level evaluation has already proved that shorter the decision tree better the explanation, similarly higher sparsity of linear model is better for an explanation. Using these understandings for evaluation of explanations would be considered as function level evaluation. Not having a human in the loop makes this evaluation method cheap in comparison to the other two methods. However, this suffers from low validity if defined proxies are not applicable for explainability required in that context.

While the above points talk about the approach of evaluation, (Carvalho et al., 2019) refer to the thesis paper from (Ruping, 2006) to discuss three goals for evaluating AI explanation. Similar concepts are supported by (Guidotti et al., 2018) in their "*Dimensions of Interpretability*" and "*Desiderata of an Interpretable Model*" sections.

- Accuracy of explanation – The explanation generated should support the prediction from the ML model. Incoherent explanations will not be useful.
- Comprehensibility – This refers to understandability of the explanation to the user of explanation. If the explanation is accurate but cannot be easily

understood by the user then that defeats the primary purpose of explainability. Point to note, whether an explanation satisfies the comprehensibility is dependent on the target user community of the explanation.

- Time efficiency – This refers to how quickly an explanation can be understood by the user. This is directly propositional to the quality of comprehensibility.

In summary, an explanation will be considered having high quality when it supports the prediction, easily and quickly understandable to a target user.

The above discussion still does not cover the metrics for evaluation that can be used in practice for selection of suitable explanation method. (Carvalho et al., 2019) suggests that there are two types of metrics for evaluating explanation.

- *Qualitative* metrics – (Doshi-Velez and Kim, 2017) listed five factors for qualitative metrics. That study had used cognitive chunks for quantification which means basic units of explanation.
    - *Form of cognitive chunks*: this refers to the type of output from the explanation method. The types are similar to result based categorisation in types of explainability techniques (section 2.2.4)
    - *Number of cognitive chunks*: This is related to the quantity of *cognitive chunks* in the explanation and is the quantity good enough considering the *form of cognitive chunks.*
    - *Compositionality*: This refers to whether the cognitive chunks are presented in the right structure/hierarchy for a user to comprehend.
    - *Monotonicity and other interactions between cognitive chunks*: This is related to the relationship between the cognitive chunks – linear, non-linear etc. Intuitively cognitive chunks with a linear relationship would be easier to understand for users.
    - *Uncertainty and stochasticity:* This refers to whether the explanation can express any uncertainty or stochasticity in the ML model which is being explained.
- *Quantitative* metrics – These set of metrics are preferred over qualitative metrics as this allows assessment of the quality of explanation in numbers

and makes the comparison of explanation method easy. However, there does not exist any single set of quantitative metrics that can work across all explainability techniques. (Honegger and Blanc, 2018) has proposed "*Axiomatic Explanation Consistency Framework"* based on functional level evaluation approach for an explanation from (Doshi-Velez and Kim, 2017). The study mentions that their framework is based on axioms as introduced by (Sundararajan et al., 2017) in their paper for *Integrated Gradients.* The study used three axioms with feature importance as explanation technique. However, these three concepts are generic enough to suit evaluation of many explanation techniques.

- *Identity*: Same feature and predicted output pair for an ML model must have the same explanations irrespective of how many times the explanation method is triggered.
- *Separability*: if the prediction form the ML model is different for two different observations then they cannot have the same explanations. An explanation should be able to justify the reason for different prediction.
- *Stability*: If a set of observations are similar, i.e. several features are the same but not all, and the predicted output is same then explanation technique should generate similar explanations. However, it would be very difficult to satisfy this axiom if the feature set is not reduced to only the required features for the prediction.

As the current study intends to experiment explanation methods on a DNN based model, the literature review covers explanation evaluation in the context of DNN separately. Post literature review, (Gilpin et al., 2019) suggested four different types of evaluation for an explanation. The study focused on assessing whether the explanation method intends to explain the *processing on data* inside DNN (e.g. Proxy methods, saliency mapping etc.), or tries to explain the representation of data inside the DNN (Role of layers, neurons etc.) or the DNN architecture itself is capable of generating explanations (e.g. Attention-based network etc.).

- *Completeness compared to the original model* – this applies to the method designed to explain the processing of the DNN. For example, a proxy model

can be evaluated based on how close is the approximation to the original ML model.

- *Completeness as measured on a substitute task* – This is applicable for both methods designed to explain processing of the DNN and representing the data inside the DNN. For example, model sensitivity generated from a salience mapping can be evaluated against derived model sensitivity from a different approach.
- *Ability to detect models with biases* - This is applicable for both methods designed to representation data inside DNN and self-explaining DNN. For example, whether the explanation method can detect model sensitivity towards a pattern where it is already known that the model should or should not have the bias.
- *Human evaluation* - This applies to all the three types mentioned above. Human evaluation can be used to evaluate explanation provided the person has the idea of how the original model is supposed to work.

2.2.6 Post-hoc AI explainability techniques

As the focus of this study is post-hoc explainability for a Deep Neural Network based model, this section will review model specific and model agnostic explainability methods that were applied on DNN models in recent past.

- **Model-specific explainability methods**

As discussed in the earlier section, model-specific explainability methods work on a specific class of models alone. However, as these use the internals of the model to arrive at interpretability, these methods are comparatively faster than model agnostic methods. This sub-section will discuss few model-specific explainability methods that got introduced for explaining DNN models.

(Simonyan et al., 2013) used saliency scores to visualize image classification models learnt using deep Convolutional Networks (ConvNets). This study proposed an understandable visualisation of ConvNet classification models that can be obtained using a single backpropagation pass through a classification convolution network. This method computed an image specific class saliency map, highlighting the areas

of the given image that discriminates the image from the other classes with respect to predicted class. This was a model-specific feature relevance explainability technique. As this method uses the absolute value of the gradient as the saliency score, that prevents differentiating between the positive and negative impact on prediction (Ancona et al., 2017).

(Zeiler and Fergus, 2014) proposed another model-specific feature relevance explainability approach to visualise the activity within the model using Deconvolutional Network (deconvnet). Aim of deconvnet was to project the feature activations back to the input feature space. The methodology in the study could demonstrate part of the image important for classification. This also shows how these visualisations can be used to identify problems with the model and use them to get better models. While this method is only applicable to CNN architectures, few other methods apply to only specific DNN architectures, e.g. Grad-CAM for CNNs or Guided Backpropagation for ReLU. These methods are not discussed further here as these methods do not apply to the current study (Ancona et al., 2017).

(Bach et al., 2015) used Layer-wise Relevance Propagation (LRP) to understand how relevance score from the output layer can be distributed to the input layer. The paper aimed to understand the contribution of a single pixel of an image to the prediction made by a classifier in an image classification task. LRP redistribute the relevance score of activated nodes in the output layer to features in the input layer via the backpropagation operation. The relevance score refers to the value calculated by the activated node in the output layer.

(Sundararajan et al., 2017) proposed a new technique named Integrated Gradients for deep networks combining the "*Implementation Invariance*" of gradients along with the "*Sensitivity"* from techniques like LRP. This technique attributed the prediction of a deep network to its inputs. This paper also highlighted desirable features of an attribution method using an axiomatic framework without which it is difficult to ascertain whether the attribution method is affected by data, network, or the attribution method itself.

(Shrikumar et al., 2017) proposed another model-specific feature relevance explainability technique, DeepLIFT, which claimed to overcome issues in former gradient based methods like Deconvolution Network, LRP, integrated gradients that could produce misleading importance scores in cases of zero or discontinuous gradients. DeepLIFT proposed computing importance scores based on the difference of the output from some 'reference' output in terms of differences of the inputs from the 'reference' inputs. This strategy was effective in the case of zero and discontinuous gradients. DeepLIFT proposed to have different considerations for positive and negative attribution of features at nonlinear functions which is more effective compared to earlier approaches. It is observed that there could be a couple of disadvantages of using DeepLIFT - (1) choosing appropriate reference input (especially for text-related tasks) would need domain expertise (2) choosing appropriate rule ("*Linear*" vs "*Rescale*" vs "*RevealCancel*") for calculating contribution scores at nonlinear functions. However, (Lundberg and Lee, 2017) proposed a new technique, DeepSHAP, based on DeepLIFT and Shapely Values (based on game theory). DeepSHAP can address the inconvenience of choosing rule ("*Linear*" vs "*Rescale*" vs "*RevealCancel*") heuristically for calculating contribution scores at nonlinear functions. Literature review could not discover an instance where DeepLIFT (Shrikumar et al., 2017) has been adapted for any text-related tasks. The original study used one image classification (MNIST data set) and one simulated Genomics classification data set to compare importance scoring.

- **Model agnostic explainability methods**

As the name suggests, these methods can be used to explain any class of models which is referred to as model flexibility by (Honegger and Blanc, 2018). The same thesis suggested that model agnostic explanations should also support explanation flexibility and representation flexibility inspired by (Ribeiro et al., 2016). Explanation flexibility refers to the ability to present the explanation most suitably to the user. This can be presented in the form of graphs or charts or even linear equation depending on how that will be understandable to target user of explanation. Representation flexibility refers to the ability to use both raw and transformed features while presenting the example. In the case of text, a fitting example of representation flexibility is to be able to use the actuals words and not be restricted

to use word embedding vectors while presenting the explanation. This subsection will first provide an overview of few model-agnostic explanation methods and then will discuss two of them in detail as those two explanation methodologies are used in the current study.

(Ribeiro et al., 2016) proposed Local Interpretable Model-Agnostic Explanations (LIME) which was a model agnostic approach to explain the predictions in an interpretable manner. The main idea of LIME was to learn a new locally interpretable (potentially linear) model around the predictions that needs interpretation. The purpose of LIME was to instil trustworthiness of a prediction. The paper also stated how explanations from LIME can be used for model selection by evaluating the generalizability of a model.

Same authors who proposed LIME extended their explanation methodology in a subsequent study (Ribeiro et al., 2018). Similar to LIME, Anchors is also a perturbation-based explanation technique; however, this does not use locally interpretable models for the explanation. Instead, Anchors use the exploration concept from reinforcement learning to generate perturbations around the individual instances that are being explained. The output of this explanation method is IF-THEN rule which is more intuitive than the locally interpretable model that LIME generates. Additionally, Anchors come with the concept of coverage which indicates the proportion of perturbation space that can be covered by the same explanation. This means Anchors can provide information regarding the other unseen observations for which the explanation will still be valid (Molnar, 2019).

According to (Honegger and Blanc, 2018), Shapley values was introduced by Lloyd Shapley in 1953. This is a cooperative game theory concept. The main objective of Shapley values is to distribute the outcome fairly amongst all players taking part in a coalition. This concept incentivises the players in the coalition by allocating more or equal reward according to their (unequal) contribution compared to the reward they would have achieved if acted independently. Based on this concept, (Molnar, 2019) discussed how this can be used as model agnostic explanation method. According to the book, for a prediction, the feature values for that observation can

be thought of as players of a game and the prediction as the outcome. Then the prediction can be explained by Shapley value assignment which indicates the distribution of the outcome to each feature value according to their role in the prediction. The Shapley value calculated for a feature value by measuring the effect on perdition by adding and removing the feature value to other feature value coalitions. This is calculated for all instances where the feature value is present. This is computing-intensive and that increases with a larger feature set. Therefore, sometimes this is managed by limiting the Shapely value calculation with a sample of possible feature value coalitions.

SHAP (SHapley Additive exPlanations) was introduced by (Lundberg and Lee, 2017) where the methodology unifying existing methods - LIME, DeepLIFT, LRP and classical Shapley values from game theory. Kernel SHAP was proposed as Linear LIME + Shapley values where SHAP could avoid the uncertainty of heuristically choosing loss function, weighting kernel and regularization term which was the case for LIME. Apart from Kernel SHAP, the paper also proposed Linear SHAP for linear models, Low-order SHAP where the number of features is less, Deep SHAP (DeepLIFT + Shapley values) specific to DNNs.

Following sub-sections will discuss LIME and Kernel SHAP in further details as the current study indents to use these two post-hoc model agnostic methodologies in the experiments.

- ***Local Interpretable Model-agnostic Explanations (LIME)***

(Molnar, 2019) summarised how to generate explanation using LIME. Intuitively LIME allows interacting with the black box ML model to understand single prediction without having any prior knowledge of training or model internals. LIME generates an explanation from the ML model predictions based on induced variations in the input data. LIME perturb the data around the prediction to be interpreted to create a new data set. LIME uses this new data set and the prediction on them to train another intrinsically interpretable model (discussed in section 2.2.4) where each observation from the new data set is weighted by the similarity to the

observation subject to interpretation. This model is a local approximation of the ML model to be interpreted.

(Molnar, 2019) has described the explanation generation from LIME as a five-step process.

i. Selection of observation for which explanation is desired
ii. Perturb the features for that observation but based on the complete dataset to create a new data set
iii. Assign weights to the perturbed samples based on similarity with original observation
iv. Build an intrinsically interpretable model using the new data sets but considering the weights
v. Explain this local intrinsically interpretable model using feature summary, model internals etc. (discussed in section 2.2.4)

The current implementation of LIME can be used with Linear regression as the interpretable model where it is required to choose the number of features upfront for generating explanations. Lesser the number of features higher the comprehensibility, higher the number of features higher the explanation accuracy. There are many techniques available, e.g. regularisation, forward / backward feature selection, to train a linear model with the exact number of features required (Molnar, 2019).

An important step in the five-step process mentioned above is the process of perturbing data. This process is different for different type of data, e.g. tabular, image, text data. The complexity of LIME lies in this step. In this section, an example of tabular data is used to discuss a few details on this step. LIME does not generate the perturbations centring around the observation of interest, instead, it uses mass centre for all the observation in the training data set. This is to have some variability in prediction from the ML model (subject to this explanation) compared to the observation of interest. LIME currently uses exponential smoothing kernel to define the scope of new data set coming out of perturbation. Exponential smoothing kernel provides a similarity measure and this is controlled by kernel width. A small kernel width means the instances in the new data set will be very similar to the instance of interest, and a large kernel width means that the perturbed observations

can be far from the instance of interest. However, choosing the right kernel width is key to the right explanation and there is no proven way to do that (Molnar, 2019).

- ***Kernel - SHapley Additive exPlanations (SHAP)***

As discussed earlier, Kernel-SHAP is proposed based on both linear LIME and Shapley values. Feature values for an observation act as the players where the prediction is the outcome. As mentioned earlier, Shapley value allows a fair distribution of outcome to the feature values according to their role in prediction. According to (Molnar, 2019), SHAP proposed new representation of these Shapely values as additive feature attribution model which is linear in nature. This allows connecting LIME and Shapley values to create Kernel SHAP.

(Molnar, 2019) has described the explanation generation from Kernel SHAP as a five-step process.

i. Arrive at all possible coalitions based on all possible feature values in data set (e.g. feature 1 – value A, feature 2 – value B, feature 3 – value C, feature N – value X). Sample from this list by randomly choosing feature values to be present in a coalition. E.g. the vector of (1,0,1,0) means a coalition of feature 1 – value A and feature 3 – value C. Create several such coalition vectors as the data set for the local regression model where the target is the prediction on the coalitions from the ML model that need explanation.
ii. Generating prediction from the ML model for explanation need the transformation of the coalition vector to the original feature space as the ML model is not trained on the coalition vector. In the coalition vector if the value is 1, then that means it can be replaced with the actual feature value; if the value is 0 then the feature value is replaced with another random value for the same feature from the data.
iii. The next step is to calculate wights based on SHAP Kernel for each of observations post transformation of coalition vector to original feature space. In this case, the calculation of weights is different from LIME. LIME calculate the weights based on the similarity to the instance of interest. In the case of Kernel SHAP, it is based on the number of coalitions present in the observations. Kernel SHAP assigns higher weights if a very high or very

low number of coalitions are present. Intuitively, this allows learning contribution of individual feature values towards prediction. If single feature value is present in coalition then it is possible to learn the contribution of presence for that individual feature value towards the prediction. Similarly, if one but all are present in coalition then it is possible to learn the contribution of absence for that "one" feature value towards the prediction.

iv. Train a weighted linear model based on the weights and data derived above. It is possible to use concepts of regularization applicable to linear model to create sparse explanations.

v. The coefficients from the linear model are consumed as the Shapley values that indicate the contribution of the feature values towards prediction.

However, there are two major disadvantages of using Kernel SHAP – (1) this is slow; therefore, it is difficult to work with Kernel SHAP if the intention is to generate Shapley values from many observations. (2) like many other perturbation based interpretability methods, this cannot take feature correlation in account as it replaces feature values with random value if that feature value is not present in sampled coalition vector (Molnar, 2019).

2.2.7 Comparison of post-hoc explainability techniques

This section will map all the reviewed post-hoc explainability techniques with the post-hoc explainability approaches mentioned in Figure 1: Post-hoc explainability approaches (Arrieta et al., 2019) and the type of explainability techniques as per section 2.2.4.

Table 2: Comparison of post-hoc explainability techniques

Explainability technique	Model-specific / agnostic	Local / Global	Approach (Arrieta et al., 2019)	Based on the result of the explanation
Saliency scores	Model-specific	Local	Visual explanation	Data points
Deconvnet	Model-specific	Local	Feature relevance	Data points
Grad-CAM	Model-specific	Local	Visual explanation	Data points
Guided Backprop	Model-specific	Local	Feature relevance	Data points
LRP	Model-specific	Local & global	Feature relevance	Data points
Integrated Gradients	Model-specific	Local	Feature relevance	Data points
DeepLIFT	Model-specific	Local	Feature relevance	Data points
LIME	Model agnostic	Local	Local explanations	Feature summary on local surrogate interpretable model
Anchors	Model-agnostic	Local	Local explanations	Data points
Shapley Values	Model-agnostic	Local & global	Local explanation	Feature summary
SHAP	Model-agnostic	Local & global	Local explanation	Feature summary on local surrogate interpretable model

2.3 AI Explainability and NLP

(Belinkov and Glass, 2019) discussed the primary motivation of interpretability in NLP tasks. In recent days DNN based model could achieve impressive performance in multiple complex NLP tasks, like machine translation, language modelling etc. In this process, human-curated NLP pipelines (rich with linguistic feature engineering) have been replaced with end to end systems driven by DNN architectures. While the goal of AI explainability

in general (as discussed in section 2.2.2) holds for NLP, the explainability in NLP task needs to address how feature creation performed using complex feature engineering steps in earlier NLP systems (pre-DNN era) are captured in DNN architectures. Earlier systems for NLP task used series for feature engineering steps to retrieve linguistic features from texts – morphological, lexical, syntactic, and semantic characteristics. These are easily understandable to a human when used as features for ML model building. In case of use of DNN for NLP tasks, as all these steps are hidden behind the complex architecture, most of the AI explainability need for NLP is focused towards understanding the transformations inside the network that is able to replace these steps.

Inspired by (Simonyan et al., 2013), (Li et al., 2016) proposed strategies for visualizing compositionality in neural models for NLP. This paper used t-SNE (van der Maaten and Hinton, 2008) visualisation to understand negations, clause compositions in Stanford Sentiment Treebank data. Using Saliency maps, it understood how much each input unit contributes to the final decision in a neural network set up for different architectures (RNN and Autoencoder) and different NLP tasks (Sentiment Classification and next-word prediction).

(Vanni et al., 2018) proposed a new deconvolution strategy, Text Deconvolution Saliency (TDS), for convolution neural network based text classification inspired by deconvolution network proposed by (Zeiler and Fergus, 2014). This study also used the t-SNE (van der Maaten and Hinton, 2008) based visualisation of compositionality as suggested in (Li et al., 2016). The aim of the study was to understand how linguistic structure is learned by ConvNet that helps to achieve high classification accuracy. The study used data sets from 3 different languages – English, French and Latin. The implementation in this paper had to improvise the deconvolution network proposed by (Zeiler and Fergus, 2014) due to difference in CNN architecture for text and images. The study demonstrated that the relevance score calculated using TDS can encode complex linguistic features like word co-occurrences and potentially other grammatical and syntactic structures. But this approach is not fit for explaining NER due to the difference in the nature of prediction task between NER and text classification.

(Samek et al., 2017b) compared LRP with sensitivity analysis (SA) for explaining the individual predictions of an AI model in terms of input features. This paper demonstrated explainability of text classification for a word-embedding based convolutional neural network trained to classify text documents from the 20News group dataset. LRP identified words supporting the predicted class as well as contradicting the predicted class. In this paper, researchers established a methodology to evaluate multiple explainability techniques. At every perturbation step, the most important words (according to SA or LRP score) are deleted by setting the corresponding input word embedding values to 0. The result confirmed quantitatively that LRP provided more interpretable results than SA. Though LRP is model specific feature relevance explainability approach, this can be applied on any DNN model. (Xie et al., 2018) used LRP for interpreting BI-LSTM-CRF based NER model which is discussed in detail later.

(Sundararajan et al., 2017) experimented their methodology for question classification task trained on WikiTableQuestions dataset. That study used integrated gradients to identify key phrases from questions that differentiate answer types (e.g. phrase ‘how many’ in question will predict answer type as Numeric, phrase ‘which film’ in question will predict answer type as String etc.). Similar to LRP (Bach et al., 2015), Integrated Gradients is model specific feature relevance explainability approach and can be applied on any DNN model. However, Integrated Gradients will have same disadvantaged as LRP (discussed later in this section) if used to interpret BI-LSTM-CRF based NER model.

According to (Volpato, 2019), ability to explain specific prediction is required for AI explainability. The blog suggests that the AI explainability in NLP lacks this ability at present. Though it is theoretically possible to design systems that generate the explanation along with the prediction, that would need manually annotated explanation which is impractical in most of the scenarios.

The alternate approach is to use Local surrogate interpretable model like LIME to explain single predictions. Rest of the section discusses similar methods of explanation.

The study in (Ribeiro et al., 2016) used two product review sentiment analysis datasets consisting of books and DVDs to analyse interpretability outcome on text classification. In

the experimental setup, the study has trained multiple algorithms for text classification including transparent models like linear regression and decision tree and non-transparent models like SVM and Random Forest. Then important features from the transparent models are used to evaluate the performance of LIME by comparing the important features extracted from explanation generated for non-transparent models. When LIME is implemented on text data the process of generating the new data set using perturbation is different from the process for tabular data (described in section 2.2.6). In the case of text, LIME removes words randomly from the original text (corresponding to the prediction that needs explanation) to come up with the texts for the new data set (Molnar, 2019).

In general, interpretations generated from local explainability approach suffer from human bias as the technique need a separate model building and human interpretation of the new model outcome. LIME specifically considered unstable as it creates a sparse linear model locally which would not be able to fit the observations efficiently if observations are not linear locally. (Villarroya, 2018) has adapted LIME for explaining DNN based NER model which is discussed in detail in the next section.

(Alvarez-Melis and Jaakkola, 2017) use the model agnostic and local interpretability concepts from LIME to explain two different sequence-to-sequence NLP tasks - English to German Machine Translation (MT) and a simple dialogue system. Due to sequence-to-sequence nature of the task, the type of explanation was expected to be different from classification tasks. The study had to adapt a novel approach as an explanation that would need to cover combinations of input and predicted tokens. To address the challenge of generating a perturbed dataset, the study used variational autoencoder (VAE) on sentences. To address the need of explanation between various input and predicted token combination, the study generated explanations as a summary of operation instead of the local interpretable model. In the task involving a simple dialogue system, the aim of the explanation was gathering insights for an imperfect black-box model. Using explanations, the study could demonstrate that the black-box model is biased towards a few words, using only partial information from input to make the prediction.

(Robinson and Yufeng, 2019) and (Li, 2019), in their blogs demonstrated the use of DeepSHAP on DNN based text classification to generate global explainability to build user

trust on the model. This suggests that Kernel SHAP can be used to generate global explanations for NER tasks as well.

2.4 AI Explainability for NER

2.4.1 Named Entity Recognition (NER)

In NLP and information extraction, NER is a sequence tagging task where each word in the sentence (i.e. a sequence of words) are tagged with entity names. Entity names are generally proper nouns, e.g. person, location, organisation etc.

Historically various types of rule-based (e.g. regular expression) and statistical models like Hidden Markov Model (HMM), Maximum Entropy, Conditional Random Field (CRF), general classification models were used for named entity recognition. All these algorithms were applicable for other sequence prediction tasks like parts-of-speech (POS) tagging (e.g. noun, verb, adjective etc.), chunk tagging (e.g. noun phrase, verb phrase, prepositional phrase). As discussed in the last section, all the above-mentioned algorithms were highly dependent on rich linguistic feature extraction pipelines. Few examples of these linguistic features are the length of the word, POS tag, presence of upper-case letters, presence of numbers, any specific business rules like the presence of a character that indicates currency (e.g. $) etc. Feature engineering of this nature makes the model tightly coupled to data and may not generalise well on a completely new data set. However, CRF is the first choice of algorithm for machine learning practitioners for sequence prediction tasks due to its transparency.

According to (Villarroya, 2018), the introduction of word embedding in (Collobert et al., 2011) was a step towards reducing manual feature engineering which helped in achieving a level of generalisation for multiple NLP tasks. However, training a multi-layer neural network based on word embedding with certain window size was not good enough to capture the context that was required for NER. The Bidirectional LSTM + CRF architecture proposed by (Huang et al., 2015) was more powerful than the multi-layer neural network architecture in capturing the context. Additionally, the CRF in the last layer was used to capture and use linguistic features, like spelling

and context features, in the sequence prediction. (Huang et al., 2015) argued that the proposed BI-LSTM-CRF architecture (see Figure 3) was able to achieve state-of-the-art performance on two sequence tagging tasks – POS and NER. It showed less dependency on word embedding as well.

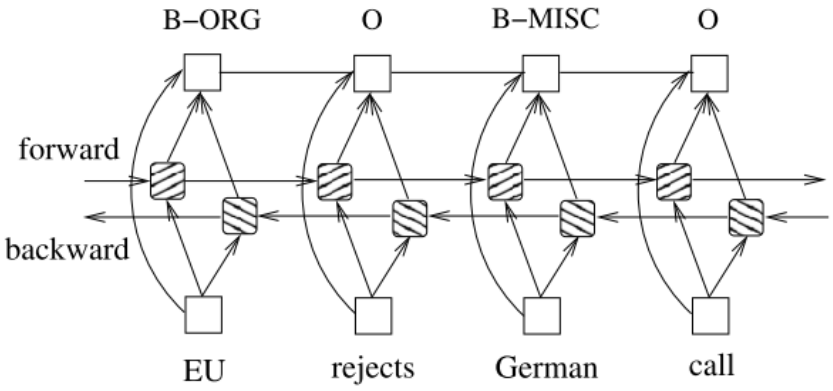

Figure 3: BI-LSTM-CRF model with linguistic features (Huang et al., 2015)

Note, Figure 3 is using IOB2 tagging scheme where beginning word of an entity is tagged as 'B-<entity name>', e.g. 'B-ORG'; any subsequent word that is part of the entity is tagged as 'I-<entity name>', e.g. 'I-ORG'. All other words that are not part of the entity are tagged as 'O'.

2.4.2 Explainable NER

Literature review suggests that AI explainability in NLP have been mainly focused on explaining predictions for classification models. In the recent past, ML explainability techniques were studied on NER models only in a couple of cases.

(Xie et al., 2018) demonstrated the use of LRP to interpret the Bi-LSTM-CRF model used in Named Entity Recognition. The primary purpose of the paper was to investigate different explainable deep learning methods for NER, implement an appropriate explainable method for the named entity recognizer based on Bi-LSTM-CRF, and explain the NER model to people having some background knowledge using the visualization. This paper used t-SNE (van der Maaten and Hinton, 2008) visualization for word embedding and used LRP on Bi-LSTM-CRF to visualize how much each word contribute to the output and how contextual words are related. This study also tried to explain the wrong NER prediction using several examples. This

paper also highlighted the need to apply other explainability techniques on NER to validate the explanation generated using LRP. The study also indicated the need for further consideration on how to use AI explainability to analyse wrong NER predictions. The result of the study lacks in two aspects - (1) interpretability for layman users and (2) interpretability of CRF layer as use of LRP was focused on the Bi-LSTM layers. LRP and any other gradient-based explainability approach will have this limitation as all are DNN specific explainability method.

In another study, (Villarroya, 2018) demonstrated AI explainability on NER task using LIME. The primary goal from the study includes generating an explanation for NER task using LIME and evaluation of explanations for entities. This paper implements NER using a Bi-LSTM and generated explanation using two different techniques – LIME and LIME-NER which is a variant of LIME for the explanation of sequence detection task.

(Villarroya, 2018) discussed that explaining NER is different from explaining classification task as in classification there would be only one predicted outcome for a single observation. In the case of NER, the predicted output is another sequence and the length of the sequence is dependent on the number of words in the input sentence. Additionally, not all part of the sentence would have the same value in explanation. In general, LIME expects the instance and probability vector corresponding to all possible classes as input for explaining classification tasks. In theory, for sequence labelling task, it is possible to pass a single word and corresponding probability vector containing probabilities for all possible entity classes to LIME for generating an explanation for each word and NER tag combination. This means it will be required to track changes in probability vector (where one element in the vector correspond to the probability of that word being a particular entity class) for each word in the input sentence from new data set generated by perturbating sentences.

Though the above approach will work, but not ideal for multi-word entities. For single word entities, the above approach will work as expected as LIME will create the local interpretable model based on the change in probabilities for the words in

the vicinity of the entity word that needs explanation. (Villarroya, 2018) argued that it is necessary to generate the explanation in entity level (irrespective of single or multi-word) and proposed couple of simple modifications in the process for generating explanation for multi-word entities.

- Post prediction of NER tags for each word in the sentence, modify the output probability vector corresponding to the number of entities. i.e. for multi-word entities, NER will provide probability vector for each word. Aggregate the probability vectors as the weighted average based on the position of the word inside the entity. This modified probability vectors for each word is then passed on to LIME for the explanation.
- In the perturbation step, it suggests perturbation of a complete entity instead of single words that may happen to be part of an entity. In that study, perturbation was achieved by zero embedding vectors. The above alteration means that while perturbing if one word from an entity is being perturbed then all the words from the entity will be set to zero embedding.

The above alterations in the process ensure that explanations can be generated in an entity level instead of explaining each word in the entity individually.

For evaluation, the study used a method named "*Area over the perturbation curve*" (AOPC) adapted for LIME-NER. The AOPC metric relies on a perturbation process similar to LIME, though they follow a different strategy for sampling of the words to perturb. This paper indicates that use of Kernel SHAP for explaining NER would be a potential future work and suggested use of the LIME-NER technique on a different data set to understand generalization of this explanation techniques for NER task.

2.5 Discussion

The literature review highlights three focus areas that need more exploration especially in the context of explanation in NER tasks - (1) choice of explainability method for NER task, (2) making explanations understandable to layman users (3) evaluation methodology for an explanation

In general, multiple reviews on AI explainability suggests that post-hoc, model agnostic explainability techniques are the future of AI explainability (Molnar, 2019). There is at least one study, (Villarroya, 2018), available where a similar technique, LIME, was used to explain NER task. However, generating meaningful explanation from LIME is subject to the right choice of the kernel width for building the local interpretable model. Use of Kernel SHAP is expected to overcome this problem and thus preferred.

A part of literature review focused on how AI explanation should be influenced by psychology. This highlights a few characteristics that human would expect in AI explanation. Two primary aspects are highlighted below.

- Generated explanations should be short (not more than one or two causes, unless regulatory requirement), should answer why question with respect to another reference point, should highlight abnormalities if present and should be understood in little time (Miller, 2019).

- Selection of specifics in the above dimensions should be dependent on the stakeholder of the explanation. Explanation generated for the expert user and layman user cannot be the same (Ribera and Lapedriza, 2019).

Study of AI Explainability on NER tasks show that there is a lack of research on generating explanation for layman users (Xie et al., 2018).

Literature review suggests that there is no consensus amongst the research community on evaluation methodology for generated explanations. Few of the studies have laid some ground rules, but still, there is an ambiguity in identifying suitable quantitative metrics for measuring the performance of explanations. While as per (Doshi-Velez and Kim, 2017; Carvalho et al., 2019; Molnar, 2019), it is important for explanation to have qualitative characteristics like accuracy, comprehensibility and time efficiency fitting to the stakeholder, explanations should also highlight model biases and shortcomings to the ML practitioner community to drive continuous improvement (Alvarez-Melis and Jaakkola, 2017; Gilpin et al., 2019).

2.6 Summary

This study has reviewed a significant number of literature including conference proceedings, journal articles, survey papers, thesis, books and blogs to understand what is

the common understanding of explainability in AI, why explainability is required in the present-day ecosystem, how explanations can be made meaningful to the users and how does this change based on users. The study looked at the different type of AI explainability techniques and when to apply which technique. It discussed the evaluation methodologies that are already laid out and how they are still not sufficient for measuring the performance of explanations. The study also looked at a few recent post-hoc AI explainability techniques for DNN models and discussed them in-depth for a couple of model agnostic techniques. Eventually, it presented a comparison amongst post-hoc AI explainability techniques in multiple dimensions.

The study noticed how AI explainability is adapted for NLP tasks and additional asks for explanations in NLP. It had studied how AI explainability was adapted for sequence-to-sequence task in NLP. Finally, it reviewed the work done on AI explainability for NER task and discussed one of them in detail where a model agnostic post-hoc local explanation technique was used.

In the last section, the study has discussed the gaps found from the literature review relating both AI explainability and NER which is the motivation for the current study.

CHAPTER 3: METHODOLOGY

3.1 Introduction

The study would like to answer how Kernel SHAP can be used to generate explanations for NER models, how explanations generated from Kernel SHAP would be comprehensible to users and how the explanation generated for NER can help improving model performance. The study will be employing statistical methods like Kernel SHAP which combines LIME and classical Shapley values from game theory for generating explanations for NER model.

Subsequent subsections will discuss all steps that will be required to address the goals mentioned above. The diagram in Figure 4 depicts the planned sequence of activities in the modelling and explanation generation phase.

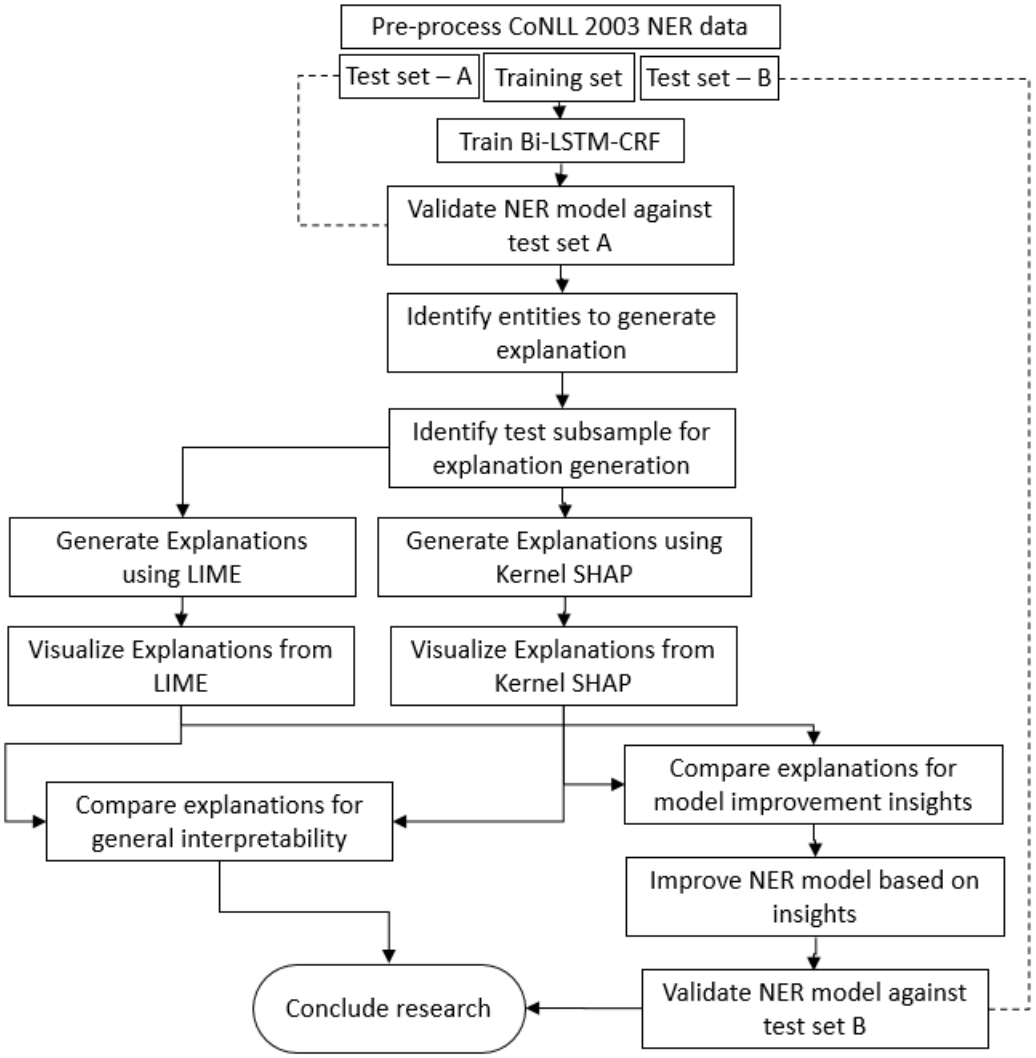

Figure 4: High-level research approach

The overall research approach can be subdivided into two broad set of activities. The first set of activities will result in the NER model that will be subject to the explanation. The first subsection will discuss the processes of building that NER model, starting from the data structure, nuances in data, transformations required in model building, setup of the NER model and model evaluation.

The second set of activities will generate the explanation of the NER model. The second subsection will lay down the blueprint for baselining explanations and evaluation of explanation. Finally, this subsection will discuss the novelty in the methodology that will address the primary goals of this study. That will discuss the generation of explanation using Kernel SHAP and evaluation of explanation to assess whether the explanation is suitable for layman users and validate whether the explanations can provide insights to improve the NER model.

3.2 Building the NER model for explanation

This section will provide the set of activities to be performed to build the NER model which is the first part of the research approach. NER model building is in the critical path for overall research objective as this will provide the base model for experimenting post-hoc explanation methods.

3.2.1 Dataset Description

This study will use the CoNLL 2003 English named entity dataset (Language-Independent Named Entity Recognition (II), 2020)[2] which is the same dataset used by (Xie et al., 2018) in their study of applying LRP on NER task. This is a public dataset containing every sentence tokenised in words. Each word present in new line along with space-separated parts of speech (POS), chunk name (e.g. noun phrase, verb phrase, prepositional phrase etc.) and named entity tags.

[2] A direct downloadable copy of this data is available at following GitHub location: https://github.com/synalp/NER/tree/master/corpus/CoNLL-2003

```
EU NNP I-NP I-ORG
rejects VBZ I-VP O
German JJ I-NP I-MISC
call NN I-NP O
to TO I-VP O
boycott VB I-VP O
British JJ I-NP I-MISC
lamb NN I-NP O
. . O O
```

Figure 5: CoNLL 2003 English named entity dataset - words

The last attribute in Figure 5 is the names entity tag. The data set contains four different entities – LOC: location, PER: person, ORG: organisation and MISC: miscellaneous. The data set follows IOB tagging scheme where all entity words are tagged as 'I-<entity name>', e.g. 'I-PER' for a person entity; only if there are two adjacent entities of the same type then 'B-<entity name>' is used for the first word of the second entity, e.g. 'B-PER'. All other words that are not part of the entity are tagged as 'O'.

Chunk/phrase name (third attribute) follows the same IOB tagging scheme. All words of a phrase are tagged as 'I-<phrase name>', e.g. 'I-NP' for Noun phrase words; if there are two adjacent phrases of the same type then 'B-<phrase name>' is used for the first word of the second phrase, e.g. 'B-NP' for the first word of a second adjacent noun phrase.

Two consecutive sentences in a single document is separated by an empty new line. The blank line shown in Figure 6 indicates the end of one sentence and the beginning of the next sentence

```
boycott VB I-VP O
British JJ I-NP I-MISC
lamb NN I-NP O
. . O O

Peter NNP I-NP I-PER
Blackburn NNP I-NP I-PER
```

Figure 6: CoNLL 2003 English named entity dataset - sentences

Multiple documents in the data set are separate by a special record as shown in Figure 7.

```
imports NNS I-NP O
. . O O

-DOCSTART- -X- O O

Rare NNP I-NP O
Hendrix NNP I-NP I-PER
```

Figure 7: CoNLL 2003 English named entity dataset - documents

Table 3 shows the size of sentences and words in training, test-a and test-b datasets.

Table 3: CoNLL 2003 English named entity dataset details

Data Set	Number of Documents	Number of sentences	Number of words
Training	946	14987	203621
Test-a	216	3466	51362
Test-b	231	3864	46435

Table 4 provides the number of tags for each data set mentioned in Table 3.

Table 4: CoNLL 2003 English named entity dataset details - no. of tags

Tags / Data set	Training	Test-a	Test-b
O	167397	42120	37902
I-PER	11128	3145	2773
B-LOC	11	0	6
I-LOC	8286	2094	1919
B-ORG	24	0	5
I-ORG	10001	2092	2491
B-MISC	37	4	9
I-MISC	4556	1264	909

3.2.2 Data pre-processing

The CoNLL 2003 English NER data set is curated for the NER task and therefore does not have any major inconsistencies. However, to do error analysis post NER model building, it will be required to add sentence index and document index that is missing

in the data set. Additionally, the data has some missing values for words, that will be handled in this pre-processing step.

- Addition of sentence index – it will be required to add an index for a sentence so that words from a single sentence can be recognised while processing. Training of a NER model is expected to consider the contextual information for a word and a context is defined by the sentence. The words are the sequence of items in the sentence.
- Addition of document index – While the training process will be unaware of the source document for any sentence, a document index will be added for audit purposes. This will help in prediction error analysis at the later stage.
- Handling missing data –
    - Post addition of sentence ID and document ID, all empty lines (indicating sentence separation) and lines starting with '-DOCSTART-' (indicating document separation) will be deleted.
    - There are close to 11% records in the training set where the word is space. This is invalid as a word. Space supposed to be the word separator. Therefore, these records will be deleted.

3.2.3 Data Transformation

Sentences are variable-length word sequences and this cannot be used as input to deep neural network model as is. Neural network models need data to be represented in numeric format and fixed length. This section will discuss the transformation steps required to achieve that. Additionally, the CRF layer in the BI-LSTM-CRF architecture (that will be used for NER modelling) will be fed with few linguistic features. This section will discuss the feature creation and subsequent transformation steps planned for the same.

- Word Embedding initialisation with GloVe (GloVe: Global Vectors for Word Representation, 2020) – The NER model will be built using a BI-LSTM-CRF architecture where the words will be the inputs to the Input layer of the DNN. Word inputs will be fed to an embedding layer.

  In theory, it is possible to use different embedding schemes. Simplest of all is the One-hot encoding based on the bag of words. The result of this embedding is an n-dimensional vector where n is the number of words in the vocabulary. This vector

would have non-zero value only for a single position and all other values will be zero (this is the vector representing only that word). This scheme suffers from high sparsity and dimensionality in embedding and does not have the notion of order in which words appear in the text.

Another simple method is the co-occurrence matrix. This takes the context of the target word in consideration as this matrix is created based on the co-occurrence of other words in the vocabulary. However, this technique also suffers from the very high dimensionality (dimension is decided by the number of words in the vocabulary in this case).

The GloVe embedding is based on the co-occurrence matrix. It is trained on non-zero entries in the word-word co-occurrence matrix and uses statistical methods to assign value based on the frequency of co-occurrence. GloVe can provide a dense, low dimensional, context-aware representation of words. This study will use pre-trained GloVe embedding to initialise the weights of the embedding layer. This pre-trained GloVe embedding is trained on Common Crawl[3] i.e. open repository of web crawl data. This embedding is based on 840 billion tokens, 2.2 million words vocabulary, cased i.e. the original case of the word - upper or lower is retained and provides 300-dimensional vectors as word representation. Size of the embedding is 2.03 GB.

- Word to word-index transformation – As input to DNN architecture will only accept numeric representation, words cannot be directly fed to the input layer, therefore each word from the vocabulary of training set will be converted to an index number. To reduce rare words and the words not present in GloVe embedding, the study will only use the words that are either available in GloVe embedding or occurs more than 10 times in the training data. This will need following adjustments to original GloVe embedding matrix.
    - Limit embedding matrix to have only the words after the reduction of the number of words in the aforementioned criteria.
    - The sequence of words in word index mapping and embedding should match.

[3] https://commoncrawl.org/

- Populate embedding with random values (between -1 and 1) for words that are present in training data but neither present in pre-trained GloVe embedding nor occurring at least 10 times in training corpus.
- Populate embedding with random values (between -1 and 1) for words that are not present in GloVe embedding but retained due to having occurrence 10 or more times in the training data.

- Variable to fixed sentence length - Sentences in the training corpus does not have a fixed length. To feed the words (in the form of word indices) to the input layer of DNN, the study will use a fixed number of words in the sequence. The strategy will be to calculate summary statistics for sentence length and to use the 3rd quartile value as the fixed sentence length for training. Any sentence having more words than the fixed length will be removed from all train, test-a and test-b data sets. Theoretically, it is possible to truncate the sentences up to a fixed length. But that will have the risk of truncating a part of an entity (for multi-word entities); therefore, decided against it. Any sentence having a smaller number of words than the fixed length will be padded with a fixed dummy word '<PAD>' towards the end till the sentence length reaches the fixed length.

- IOB to IOB2 tagging – IOB tagging scheme does not provide a clear and consistent view for the number of entities and chunks in the data set. Both NER and Chunk tags will be converted from IOB to IOB2 tagging scheme. I.e. beginning word of an entity will always be tagged as 'B-<entity name>', e.g. 'B-PER'; any subsequent word that is part of the entity will be tagged as 'I-<entity name>', e.g. 'I-PER'. All other words that are not part of the entity are tagged as 'O'. In case of chunk name, beginning word of a phrase will be tagged as 'B-<phrase name>', e.g. 'B-NP' for the beginning of Noun phrase; any subsequent word that is part of the same phrase will be tagged as 'I-<entity name>', e.g. 'I-NP'.

- Spelling related feature creation – Feature creation step in (Huang et al., 2015) mention several spelling related feature generations so that the features can be directly fed to the CRF layer bypassing the Bi-LSTM layers. Below is the subset of spelling related features from the paper that is planned to be used in this study.

- number of characters in the word
- is the first letter of the word in upper case
- are all letters of the word in upper case
- are all letters of the word in lower case
- is any letter in the middle of the word in upper case
- does the word contain both letters and digits
- does the word contain punctuation
- word pattern – all upper-case letter replaced by ‘A’, all lower-case letter replaced by ‘a’, all digits replaced by ‘0’, all punctuations replaced by ‘-’
- word pattern summarisation - all consecutive upper-case letter replaced by ‘A’, all consecutive lower-case letter replaced by ‘a’, all consecutive digits replaced by ‘0’, all consecutive punctuations replaced by ‘-’

The study will use 5 words prior and 5 words next to the target words for spelling related features.

- Context related feature creation - Feature creation step in (Huang et al., 2015) mentions the use of POS and CHUNK as context-related features. Same is planned to be used in this study as well. The study will use 5 words prior and 5 words next to the target words for context-related features.

- One-hot encoding of spelling and context related features – Though most of the spelling related features are either Boolean (e.g. is the first letter of the word in upper case) or numeric (e.g. the number of characters in word), there are two spelling related features that are string - word pattern and word pattern summarisation. Similarly, all context-related features are string. To feed these to DNN, one-hot encoding will be performed on these string features. This means a new column will be created for each unique value of these features and will be populated with 1 when the original feature value matches the value represented by the column, or 0 otherwise.

- Standardise numeric features – One of spelling related feature, number of characters in a word, is of the numeric datatype. Therefore, that feature needs standardisation by removing the mean and scaling to unit variance before providing as input to the Bi-LSTM-CRF model. I.e. arithmetic mean of all values for that feature in the

training set will be subtracted from the value and then that will be divided by the standard deviation of all values for the same feature in the training set.

3.2.4 Bi-LSTM-CRF for NER model building

The study shall build a Bi-LSTM-CRF model for NER tag prediction as proposed in (Huang et al., 2015). The DNN architecture will have the following layers in the mentioned sequence starting from the Input layer

- Primary Input – This will be the initial input layer with dimension same as the fixed number of words per sentence. All words of a single sentence will be provided as input at once. The values at each position will be set as the word index according to the word present at that position in the sentence.

- Embedding – This is the first hidden layer in the network. The functionality of the layer would be to transform the sparse representation of the words to a dense, context-aware representation. As discussed earlier, weights in this layer will be initialised by the modified GloVe embedding wights based on words retained from the original vocabulary. The output of this layer will be a 300-dimension (as chosen GloVe embedding of the same dimension) vector of each word. Though the weights of this layer will be initialised, the weights will be adjusted while the model is learnt as part of the training on the specific NER task.

- Bidirectional LSTM – The study plans to use Bidirectional Recurrent Neural Network (RNN) architecture with Long Short-Term Memory (LSTM) cells.
    - RNN – Traditional neural networks deal with every observation individually. I.e. it cannot use learned information from past elements of a sequence as input to the prediction for subsequent elements. In the context of NER, it will be required to take the learning form contextual words in consideration as NER tag for a target word would most likely be dependent on context words (e.g. for an entity which is more likely to be multi-word, B tag will always precede I tags). This is solved using RNN architecture as RNN allows the flow of information not only to the next layer, also to the same layer while dealing with the next element in the sequence. Figure 8

represents a single RNN layer (there could be many RNN layers in the complete network) (Understanding LSTM Networks -- colah's blog, 2020).

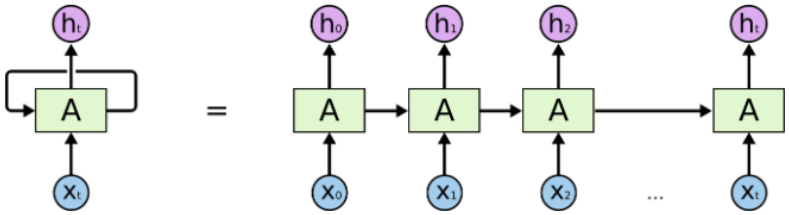

Figure 8: Single neuron in single RNN layer (Understanding LSTM Networks -- colah's blog, 2020)

$X_0, X_1, \ldots, X_t$ in Figure 8 are a sequence of inputs from different timestep 0 to t, e.g. words from a sentence that input to single RNN layer where each word is represented by embedding vector. While $h_0, h_1, \ldots, h_t$ are non-linear activation outputs, it is dependent on both current input from the last layer and output from the same layer for previous elements from the sequence. This can be represented using the following equations.

$$z_t^{(l)} = W_F^{(l)} h_t^{(l-1)} + W_R^{(l)} h_{t-1}^{(l)} + b^{(l)}$$

$$h_t^{(l)} = f^{(l)}(z_t^{(l)})$$

There are two sets of weights. W_F are the feed-forward weights (weights between two hidden layers) and W_R are the weights between two timestamps in the same layer; i.e. each neuron at timestamp 't' in layer 'l' with every neuron at timestep 't+1' in the same layer 'l'. z_t is dependent on both activation from the previous layer 'l-1'at timestep 't', i.e. $h_t^{(l-1)}$, and the activation in the same layer 'l' at the previous timestep 't-1', i.e. $h_{t-1}^{(l)}$. $f^{(l)}$ indicates the non-linear activation function, e.g. tanh, sigmoid etc.

Theoretically, RNN should be able to handle long term dependencies. However, in practice if the dependencies are long term, RNN suffers from Vanishing Gradients, i.e. weight matrices become negligible to have any effect on target output in longer timestep differences (Bengio et al., 1994). This is solved by LSTM units (Hochreiter and Schmidhuber, 1997). However, the network follows RNN architecture.

This study will use standard many to many architectures where both input and output of RNN will be sequences (There could other types of RNN architectures, e.g. encoder-decoder, many to one – can be used in next word prediction, one to many – can be used to in generations).

- LSTM – LSTM cells are designed to handle long term dependencies. LSTM cells have memory in cell state that allows RNN to retain information from elements seen earlier. There is gating mechanism that helps in deciding how much memory to be discarded form previous time steps and how much to be overwritten from current timestep and how much to be provided as output to next time step (Hochreiter and Schmidhuber, 1997). However, LSTM comes with a disadvantage over simple RNNs; the number of parameters to be trained increases multi-fold for LSTMs (four folds precisely)
- Bi-LSTM – Availability of complete text at the time of predicting NER, allows the study to use Bidirectional RNN with LSTM layers for the training of NER model. This means the sequence of words will be feed to the RNN in forward as well as backward order. As the network will have visibility of the words following the target word for NER, it would use the learning from subsequent words as well. Figure 9 represents a bidirectional LSTM network.

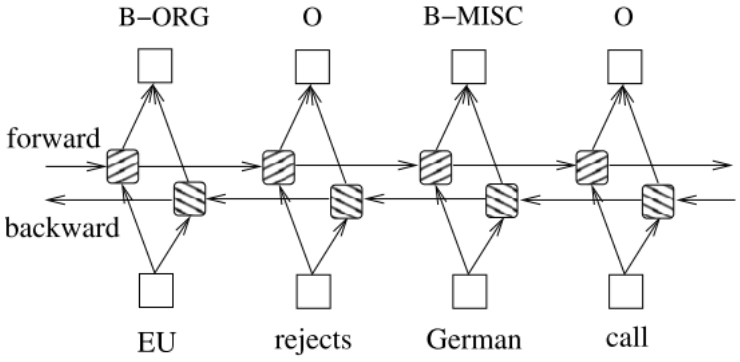

Figure 9: Bidirectional LSTM network (Huang et al., 2015)

This too comes with a disadvantage over simple RNNs; the number of parameters to be trained increases two-fold for Bidirectional RNN.

- Auxiliary Input – This study will use spelling and context related features as an additional input to the network. These features from 5 words prior and 5 words next of the target word will be concatenated to the output of Bi-LSTM layer as suggested by (Huang et al., 2015). The dimension of the additional feature matrix will be (fixed number of words decided for a sentence) x (number of created features). The output dimension of Bi-LSTM layer will have the fixed number of words decided for a sentence in one axis; the concatenation will happen in that axis. Concatenated data will be provided as input the CRF layer.

- CRF – This is the final layer in the network. Conditional Random Field is very similar to the log-linear algorithm which models the conditional probability of output sequence (BIO tag in case of NER) given the input sequence (i.e. words in a sentence). CRF uses feature functions instead of the raw input sequence alone to understand the relation between target tag and features extracted from words and the other NER tags predicted for previous words. (Huang et al., 2015) argued that use of CRF in the final layer can improve NER prediction accuracy as it can relate the precited tag in sentence level to arrive at optimum tag sequence.

3.2.5 Model evaluation

The results from the Bi-LSTM-CRF model will be validated against "test-a" dataset – there are two test sets available with CoNLL-2003 English NER dataset (Language-Independent Named Entity Recognition (II), 2020).

- All data pre-processing and transformation steps will be performed on test sets as well. All pre-processing steps will be performed as described in section 3.2.2. The transformation steps will be as follows.
    - Word to word index transformation – word index transformation will be based on the word indices defined for training data set. i.e. any new word in test set will be assigned the index of the unknown word.
    - Word embedding – at the time predicting NER tag on test data, the learned weights in the embedding layer will be used as part of the model predict function.
    - Feature creation - All spelling and context related features will be created as described in section 3.2.3
    - One-hot encoding of textual spelling and context related features – this will be performed based on the one-hot encoding scheme learned during training data transformation.
    - Standardisation of numeric spelling related feature – this will be performed based on the standardisation scheme learned during training data transformation.

- The initial NER model that will be built to generate explanations will be evaluated on “test-a” dataset using the sequence F1 score. Accuracy will not be appropriate performance metrics as the proportion of ‘O’ tags (words not a part of an entity) are much higher than the number of entities in a sentence. Sequence F1 (Hironsan, 2019) is an adaptation of the F1 score that is used as performance metrics for classification tasks (F1 score is the harmonic mean of recall and precision). Sequence F1 score will be more appropriate than F1 score for entity tags as the accuracy of an entity tag being predicted correctly is defined by both entity tag value, and start and end position for the predicted tag. Sequence F1 provides a framework to calculate the F1 score for entities as a whole instead of calculating the F1 score for each word. F1 calculation for each word individually would be misleading especially in cases where a part of the entity is tagged incorrectly for multi-word entities.

Once a reasonably accurate NER model is achieved by performing the activities mentioned in this section, preparation for next and primary stage of the study will be complete. This model will be the subject of explanation using the chosen post-hoc, model agnostic, local explanation methods. The following subsection will discuss the methodology for applying these explanation methods on the NER model and evaluation metrics for the explanation in details.

3.3 Explaining the NER model

The objectives of the study are to generate explanations of NER predictions using Kernel SHAP and evaluate whether the explanations are better than LIME-NER with respect to (1) ease of understanding and (2) ability to provide insights that can help to improve the NER model. The following subsections will discuss the novelty in the proposed methodology that will help in achieving the objectives mentioned.

3.3.1 Explanation generation

Before applying post-hoc explanation methods, the study will identify entities and observations for generating explanations. Based on the sequence F1 score, one best-performing entity and one worst-performing entity will be selected for generating explanations. Explanation generation needs extensive computing, especially for huge

datasets. Therefore, in most of the scenarios, explanations will be generated when the NER prediction probability is more than 0.95. This would mean the study will generate explanations where the NER model is confident about its prediction but will cover both cases where prediction is accurate or incorrect with respect to the ground truth.

Literature review suggests that there are at least four post-hoc, model agnostic, local explanation methods that have been developed in recent past. This study chooses LIME (Ribeiro et al., 2016) for creating a baseline explanation so that the proposed method of explanation can be compared against that. The reason for choosing LIME is the availability of the knowledge from the earlier implementation of LIME for NER model by (Villarroya, 2018).

This study proposed use of Kernel SHAP as the explanation method for NER inspired by the recommended future work in (Villarroya, 2018). The literature review also suggests other advantages of using Kernel SHAP as explainability method. This explanation method connects LIME, the baselining method, with Shapley values that has a strong foundation in game theory. It is still better than Shapely values as Shapely values cannot use a subset of features to generate selective explanations that are important for human understandability (discussed in section 2.2.3). Kernel SHAP can achieve that due to its linkage with LIME. It is also evident that Kernel SHAP is better than LIME with respect to methodology as it addresses the uncertainly in heuristically choosing the weights for observations in perturbed data (that is used in locally interpretable model building). Additionally, Kernel SHAP can provide a contrastive explanation that is important for human understandability (discussed in section 2.2.3). It can provide both local and global explanations. Local and global explanations generated using Kernel SHAP can have desired consistency due to use of Shapely Values as the basis of generating the explanations. Using separate methods for local and global explainability lacks this consistency (Molnar, 2019).

- **Baselining explanation using LIME-NER**

The study will use LIME-NER as suggested by (Villarroya, 2018) based on the original LIME from (Ribeiro et al., 2016). This will create a baseline explanation for comparison with the preferred method of explanation as per this study.

- The NER model will provide a probability vector for each word in a sentence and the size of the probability vector will be the same as the number of possible predicted tags. In this study, the size of the probability vector will be 9 having the probability values for O, B-PER, I-PER, B-ORG, I-ORG, B-LOC, I-LOC, B-MISC, I-MISC
- This probability vector for a predicted entity will be post-processed to create a probability vector having aggregated probabilities for each entity. I.e. each word will have post-processed probability vector of size 5 having the probability values for O, PER, ORG, LOC, MISC. The aggregation of probabilities from B and I tag to entity-level probability will be dependent on the type of the entity.
    - O tags - This will be the average of predicted probability of O for all the words in the predicted entity.
    - Entity tags – This will be the average of predicted probability of B tag for the first word and predicted probability of I tags for the subsequent words. This will be done for all the entities (except 'O') individually.

  For example, if there is a sentence 'word1 word2 word3 word4' and if word2 has the highest predicted probability for B-MISC and word3 has the highest predicted probability for I-MISC (i.e. word3 word4 is predicted as MISC entity) then for generating the explanation of MISC entity prediction for 'word 2 word 3', prediction probabilities will be aggregated in following way.
    - O entity – average of prediction probability of 'O' for word 2 and word 3
    - MISC entity – average of prediction probability of 'B-MISC' for word 2 and 'I-MISC' for word 3.
    - LOC entity – average of prediction probability of 'B- LOC' for word 2 and 'I- LOC' for word 3.
    - ORG entity – average of prediction probability of 'B- ORG' for word 2 and 'I- ORG' for word 3.
    - PER entity – average of prediction probability of 'B-PER' for word 2 and 'I-PER' for word 3.
- This post-processing function will be built as a wrapper around the NER model prediction function so that it can output the entity level probability vector for

each word in a sentence. This function will be passed to LIME text explainer function for creation of the locally interpretable linear model.

- There is a sampler function from LIME that controls the perturbation. This will need to be configured specifically for the NER task.
    - The configuration should ensure that the words from a single entity should be perturbed together and not as individual words
    - Notion of the sequence of words is retained while processing
- The explanations will be generated as multiclass variable importance plots – this will help to visualise the positive and negative impact of different features for each pair of possible entity classes.

- **Proposed explanation using Kernel SHAP**

In the proposed method of using Kernel SHAP for NER prediction, the following steps will need to be performed to use the method for NER. As Kernel SHAP has not been applied for NER in earlier studies, the steps discussed here will be the novelty of the study.

- Post-processor function as the model predict function- As suggested by (Villarroya, 2018) for LIME NER. This function will need to provide a wrapper for the NER model prediction function. This function will transform the predicted probability vector for each word having probabilities for O, B-PER, I-PER, B-ORG, I-ORG, B-LOC, I-LOC, B-MISC, I-MISC to have aggregated probabilities for each entity. I.e. each word will have post-processed probability vector of size 5 having the probability values for O, PER, ORG, LOC, MISC. This function will be passed to SHAP's Kernel explainer as the model to be explained. This will ensure that the explanations are generated in entity level, not for individual B and I tag.
- Creating background data as the baseline – This data set is used by SHAP to create the perturbed data set for building locally interpretable model. As discussed in section 2.2.6, in the perturbed coalition vector (indicating presence or absence of feature value combination for a perturbed observation), absence of a feature value is filled with the other values of the same feature from the data set provided. For smaller problem set it is possible to provide complete training

data for this purpose. But for a problem of this size, it is recommended to use single reference values (Explainers — SHAP latest documentation, 2020). Therefore, the study will use dummy word corresponding to unknown words and corresponding index (discussed in section 3.2.3) as the background reference data in perturbation

- Kernel SHAP explanations will be visualised using three different types of plots.
    - Force plot - This helps in visualising features contributing in pushing the model output from the base value (in case of NER base value will be determined based on the prediction of a sentence having all dummy words that are used to represent words outside GloVe vocabulary) to the model output.
    - Summary plot – This helps in visualising the mean absolute value of the SHAP values for each feature on all possible model outcome. This indicates the unsigned magnitude of importance for each feature for the final prediction outcome.
    - Multioutput decision plot – This helps in comparing the influence of each feature on every possible model outcome in a step-wise manner.

  However, there would be a need for post-processing the explanation before use of these SHAP visualisations. The features for the trained NER model are identified by the word position in a sentence and words will be different in a single position across multiple sentences. It will not make sense to visualise feature influences aggregated on word position. Therefore, the study will transform the word positions to original words from the sentence but will use the additional spelling and context related features as is. These processed feature influence values will be used to explain the NER model locally. As the spelling and context related features are directly interpretable to the human eye, this is expected to serve as a good medium of understanding for layman users.

3.3.2 Explanation evaluation

This study will perform function level evaluation of explanation due to the high cost and effort required for application-level and human level evaluation of explanations as discussed in section 2.2.5.

- **Qualitative evaluation**

  As the objective of the study is to understand whether Kernel SHAP generated explanations are more acceptable to the human user compared to LIME-NER explanations, the qualitative evaluation will focus on overall explanation accuracy and comprehensibility and time efficiency. These can evaluate the explanations based on the characteristics of a good explanation for the human user.

  - Accuracy of explanation will be measured by observing whether explanations can answer contrastive questions correctly. The instances for this assessment will be chosen based on the correct prediction of entities having probability scores very close to other entities. The steps for selecting the right instances to demonstrate this quality will be as follows
    - Find out mean and standard deviation from the distribution of entity-level prediction probabilities where prediction is accurate with respect to ground truth
    - Find out instances where prediction probability for one entity is within one standard deviation of prediction probability of another entity.
    - In theory, this will identify the instances where prediction probability for two entities are very close. E.g. the predicted probability for PER and LOC is very close for a word. In such cases, the explanation model should be able to explain why it has chosen one entity over another in prediction.
  - Time Efficiency will be understood by measuring the number of explanation elements (i.e. number of cognitive chunks as discussed in section 2.2.5) required to convey the understanding. However, the presence of few spelling and context related features in explanation can avoid the need for looking at raw word features. This will reduce the number of explanation units, but may not be intuitive for layman user. Therefore, there will be a need to use a weighting scheme while calculating the number of explanation units presented by the method. An indicative weighting scheme can be as follows
    - Some of spelling and context related features that are intuitive in nature will have lower weights than raw words, e.g. is the first letter

of the word in upper case, are all letters of the word in upper case, POS tag etc. (Type A)

- Some of spelling and context related features that are complex in nature will have higher weights than raw words, e.g. word format, word format summary, chunk tag etc. (Type B)
- Raw word features will have unit weight. (Type C)

Then this metric will be formulated as following based on the features provided in generated explanation.

$$TE_{ex} = -\frac{1}{N} * \left(\sum_{i=1}^{N} \sum_{t \in \{Type\ A,B,C\}} n_{i,t} * w_t\right) \ldots$$ *Eq(1): Time efficiency metric*

Where TE_{ex} is Time Efficiency of explanation, N is the number of instances for an entity for which time efficiency is being calculated, n_t is the number of features in Type t for single such entity and w_t is the weight for Type t. Higher the value (i.e. less negative) of the metric, better the time efficiency.

- Comprehensibility of explanation will be observed based on identity, separability and stability of explanation as discussed in section 2.2.5. This will be demonstrated based on similarity and dissimilarity of words featured in explanation based on the output of the embedding layer (that was initialised using GloVe pre-trained word vector and learned as part of NER model building). The steps would be as follows
  - If the explanation is provided using words form the text, then understand word embedding from the output of the first hidden layer in Bi-LSTM-CRF architecture explained in section 3.2.4. – this is the embedding layer
  - The study will visualize N top words in explaining the same entity across instances. The number of similar words (more is better) in explanation of multiple instances of the same entity will be used to quantify two of qualitative characteristics namely the identity and stability.
  - Similarly, the study will visualize N top words in explaining different entities across instances. The number of matching similar words (less is better) in explanation of multiple instances of different entities will be

used to quantify the qualitative characteristics namely separability. N will be chosen based on experiments performed.

- T-distributed Stochastic Neighbour Embedding (t-SNE) (van der Maaten and Hinton, 2008) model will be used to visualize the word similarities. t-SNE is a non-linear method to reduce dimension for high dimensional data. t-SNE converts the similarities of data points to probabilities and use joint probability to represent the similarity between data points. The t-SNE model will be trained using the trained word embedding weights from the embedding layer. t-SNE will represent 300-dimension words embedding vector in 2-dimensional space. In t-SNE transformed 2-dimensional space, similar words will be placed in proximity compared to dissimilar words.

It is necessary to note that the qualitative evaluation of explanation will be comparative in nature. The study does not intend to provide any evaluation of explanation in absolute terms, i.e. it will not generate any score that can be compared in a different experimental setup.

- **Quantitative evaluation**

The study will use "*Area over the perturbation curve*" (AOPC) adapted for LIME-NER by (Villarroya, 2018) to quantitatively measure the explanation quality for both LIME-NER and Kernel SHAP. AOPC was proposed by (Samek et al., 2017a) for quantitative comparison of heat maps generated from multiple explainability technique (Sensitivity Analysis, LRP and deconvolution network) on multiple image classification datasets. (Villarroya, 2018) adapted AOPC for NER explainability.

- As suggested by that study, for correct NER predictions, AOPC will be calculated by perturbing in *most relevant first (MoRF)* method. The features from ground truth class explanation will be arranged in the descending order of relevance score (This is feature importance scores for the locally interpretable model for LIME and SHAP values for Kernel SHAP). In the case of MoRF, features will be removed from start to end from this ordered list.

$$x_{MoRF}^{(0)} = x$$

$$1 \geq k \geq L : x_{MoRF}^{(k)} = g(x_{MoRF}^{(k-1)}, i_k) \quad \ldots \qquad \textit{Eq(2): MoRF}$$

Eq(2) explains the perturbation technique which indicates that $x_{MoRF}^{(0)}$ is set to original sentence without any perturbation. Then perturbed sentences are recorded after each feature removal where L is the total number perturbation iteration, g is the function that removes the word from $i_k$index in the perturbed sentence from the previous iteration (i.e. $x_{MoRF}^{(k-1)}$).

$$AOPC = \frac{1}{L+1}\left[\sum_{k=0}^{L}\left\{f_{gt}\left(x_{MoRF}^{(0)}\right) - f_{gt}\left(x_{MoRF}^{(k)}\right)\right\}\right]_{p(x)} \quad \ldots \qquad \textit{Eq(3): AOPC with MoRF}$$

Once all perturbed sentences are recorded as per Eq(2). AOPC will be applied as per Eq(3). In this equation, $f_{gt}$is the function for retrieving NER prediction probability for the ground truth entity class. $[]_{p(x)}$ indicates that the value of expression inside this will be averaged for all instances selected for this metrics calculation. i.e. the value of AOPC will be a vector of length L where each element will be average of prediction probability difference (across all instances) between prediction probability without perturbation and prediction probability after k^{th} feature removal.

In summary, NER prediction probability score for the ground-truth class will be calculated for an entity after removing most relevant word (with respect to ground truth explanation) from sentences iteratively as per the LIME-NER and Kernel SHAP scores separately. The word will be removed by setting the embedding values corresponding to the dummy word that is used to represent words not present in pre-trained GloVe embedding. This will be performed on all selected observations. NER prediction probability score for the ground-truth class will be averaged in each iteration of word removal.

- For incorrect NER predictions, AOPC will be calculated using *least relevant first* (LeRF) perturbation process which is intuitively reverse of MoFR. In case of LeRF, features from ground truth class explanation will be removed from end to start from the list of arranged words in descending order of relevance scores.

$$x_{LeRF}^{(0)} = x$$

$$1 \geq k \geq L : x_{LeRF}^{(k)} = g(x_{LeRF}^{(k-1)}, i_L + 1 - k) \quad \ldots \qquad \textit{Eq(4): LeRF}$$

LeRF in Eq(4) has only one difference from MoRF in Eq(2). The features are perturbed based on ascending order of relevance score from generated explanations of ground-truth class.

$$AOPC = \frac{1}{L+1}\left[\sum_{k=0}^{L}\left\{f_{gt}\left(x_{LeRF}^{(0)}\right) - f_{gt}\left(x_{LeRF}^{(k)}\right)\right\}\right]_{p(x)} \quad \cdots$$ *Eq(5): AOPC with LeRF*

AOPC in Eq(5) is same as AOPC in Eq(3) after perturbed sentences are achieved using LeRF as per Eq(4).

I.e. the only difference from the earlier method will be that NER prediction probability score for the ground-truth class will be calculated per entity after removing least (instead of most) relevant word from sentences iteratively as per the LIME-NER and Kernel SHAP scores separately.

It is expected that the initial steepness of the AOPC achieved by implementing the above strategies will be higher for the better NER explanation technique (LIME-NER vs Kernel SHAP). The relevance of AOPC with MoRF and LeRF for accurate and inaccurate predictions is summarised in Table 5.

Table 5: AOPC metric with MoRF and LeRF methods for accurate and inaccurate predictions

	AOPC+MoRF	AOPC+LeRF
Accurate prediction	Higher initial steepness will indicate the order of features in explanation is aligned to order of features' influence in predicting true class	Higher initial steepness will indicate more contribution from least important features, thus less redundancy in features used for explanation
Inaccurate Prediction	Higher initial steepness will indicate most influencing features in explanation for ground truth entity class having a higher influence on true entity prediction	Higher initial steepness will indicate the least important features in explaining ground truth entity class having higher influence in predicting true entity class. This is only expected for incorrect predictions.

It is also expected that variation in AOPC will be higher for the better NER explanation technique (LIME-NER vs Kernel SHAP) as features used for the

explanation by the better technique will have more power to change prediction output.

- **Model improvement insights based**

  The study intends to compare the insights from explanations generated from the two methods based on the ability to detect models with biases. This will help the study to evaluate the quality of the model improvement insights by implementing them to revise the NER model.

  - Analysis of explanations generated for multiple entities will be able to highlight unexpected biases in prediction which can be considered as flaws.
  - The plan for the study includes establishing whether model improvement insights are valid by improving the earlier NER model based on insights gathered form explanations. Once the NER model is updated based on the insights, old and new NER model will be validated on the test-b dataset by comparing entity recognition performance using the sequence F1 score.

3.4 Summary

This study indents to covers two high-level tasks to achieve its aim and objectives. The first task is the NER model building using state-of-the-art Bi-LSTM-CRF architecture. The first part of the section has provided data description including its limitations and process for cleaning them for CoNLL 2003 English NER data set.

The section discussed the detail of various feature transformations that are planned before modelling. Transformations include use of GloVe pre-trained word embedding, NER tag transformations from IOB to IOB2 format and linguistic feature creations. This section discussed the embedding, Bidirectional LSTM and CRF layers of Bi-LSTM-CRF architecture in details and talked about the method of evaluating NER models using sequence F1 score as performance metrics which is an adaptation of general F1 score used for classification tasks.

The second part of this section describes details of explanation generation. The study will first generate explanation using LIME-NER from the previous study on a similar subject; this is an adaptation of original LIME. This section describes the post-processing required on the NER

predictions to make use of LIME-NER and AOPC as quantitative metrics for evaluation of explanation generated from LIME-NER.

While the explanations generated from LIME-NER will serve as baseline explanation for the NER task, the novelty of this study is using Kernel SHAP in explaining the NER task. This section proposed the adaptations required to use Kernel SHAP for explaining NER tasks. Finally, the section discussed the method to derive qualitative and quantitative metrics for the explanation which is less explored in literature. The qualitative metrics of explanation will have the primary focus in the study as it is aligned to the goal of providing explanation of NER prediction to layman users. The study also discussed a method of evaluation for explanations based on its ability to provide insights regarding the shortcomings of the NER model.

CHAPTER 4: ANALYSIS AND DESIGN

4.1 Introduction

This section will discuss the different experiments performed as part of the study. This will cover the detailed steps in data preparation, NER model building, NER model validation in the first subsection. Then this will discuss process followed for the generation of explanation using both LIME and Kernel SHAP, qualitative and quantitative validation metrics for explanation and comparison between LIME and Kernel SHAP explanations.

4.2 Data Preparation

This section will discuss data pre-processing and data transformation related steps in details. Figure 10 depicts the steps that are performed in sequence to prepare data for model training. CoNLL 2003 English named entity dataset (Language-Independent Named Entity Recognition (II), 2020) comes with three different datasets. Training data set is used to train the NER model and test-a dataset is used to validate the model.

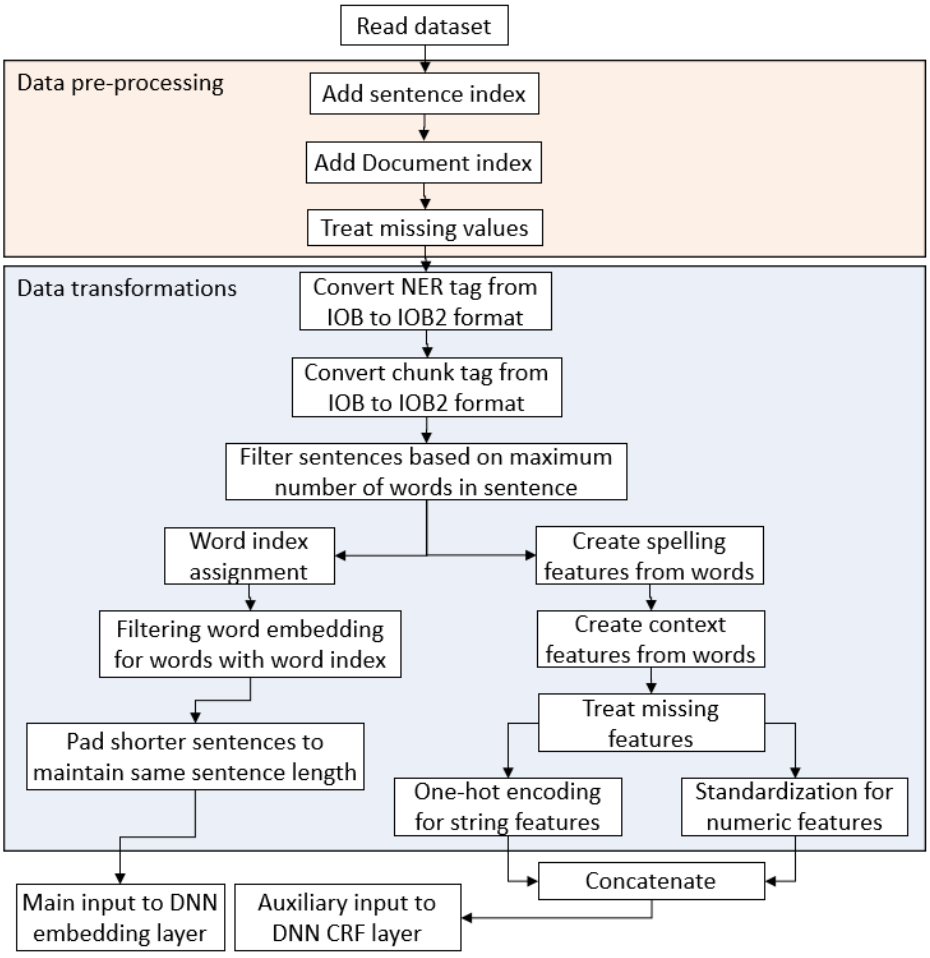

Figure 10: High-level steps in data preparation for training data

4.2.1 Data pre-processing

This section will discuss the initial pre-processing steps that are performed to add following structural attributes and treat missing values.

- Addition of sentence index – As discussed in section 3.2.1, the CoNLL 2003 English NER data set has each word in a new line. Words from two consecutive sentences are separated by one empty new line (see Figure 6). Based on this structure of the data, all words of a single sentence are assigned with a sentence index that is of numeric data type.
- Addition of document index – As per the structure of the data set, sentences from two different documents are separated by a special word '-DOCSTART-' (see Figure 7). Based on this keyword, sentences for a single document are assigned with one document index that is of numeric data datatype.
- Handling missing data – after adding sentence and document indices, missing values are treated in the next step.
    - 14987 empty lines (indicating sentence separation) and 946 lines starting with '-DOCSTART-' (indicating document separation) are deleted.
    - 11% records in training set, where the word is space, are deleted.

4.2.2 Data Transformation

This section will discuss all data transformations and feature creations performed before NER model building.

- IOB to IOB2 tagging – Both NER and Chunk tags are converted from IOB to IOB2 tagging scheme as discussed in section 3.2.3. This has revised the number of B and I tags in all three data sets as shown in Table 6 and the number of B-tags will represent the actual count of such named entity.

Table 6: Revised number of named entity tags

Tags / Data set	Training	Test-a	Test-b
O	87243	20121	21261
B-PER	4483	1008	1001
I-PER	2998	675	688
B-LOC	4789	1180	1149
I-LOC	590	156	170
B-ORG	4484	839	1184
I-ORG	2168	356	470
B-MISC	1716	460	396
I-MISC	704	193	137

- Variable to fixed sentence length – To feed the sentences in the DNN model, it is required to have a fixed length. As per the method discussed in section 3.2.3, this is achieved by filtering the sentences based on the 3rd quartile value for the number of words in all sentences in the training data set. Calculated summary statistics for sentence length are shown in Table 7.

Table 7: Summary statistics for the number of words in sentences in the training corpus

Minimum	1
1st quartile	6
Mean	14
Median	10
3rd quartile	22
maximum	113
Standard deviation	11

As per the summary statistics, approximately the 3rd quartile value, i.e. 25 words is used to remove lengthier sentences from train and test data sets. This step will reduce the number of sentences and numbers of words form all the data sets. The number of documents, sentences, and words in the reduced data set is reported in Table 8.

Table 8: Revised number of documents, sentences, and words

Data Set	Number of Documents	Number of sentences	Number of words
Training	946	11289	108975
Test-a	216	3250	50719
Test-b	231	3453	46014

Any sentence having a lesser number of words than 25 will be padded with a fixed dummy word '<PAD>' towards the end till the sentence length reaches the fixed length. This step is discussed in more detail while discussing the steps for the word to word index transformation.

- Word to word-index transformation – To convert words to index numbers (that can be used as input to DNN), only the words that are either available in GloVe embedding or occurs more than 10 times in the training subsample are considered. All these words are assigned an index value. This step results in the following.
    - Total number of unique words in original training dataset: 23624
    - Number of original words that are used after the word to word index conversion: 21402
    - Per cent of original words that are used: 90.59%

  A subset of the original GloVe embedding matrix is created that can be used to initialize the embedding weights for the embedding layer in the DNN. Following steps are performed to create the subset.
    - GloVe embedding for only those words are retained that got assigned an index in the previous step
    - A new word is introduced in the word index– '<UNK>' to represent (1) the words that are present in training data but neither present in pre-trained GloVe embedding nor occurs at least 10 times in training corpus. (2) words that are not present in GloVe embedding but retained due to having occurrence 10 or more times in the training data. There are only 19 such words (out of 21402) that will be imputed with random embedding. Populated embedding values with the random number (between -1 and 1) for the word '<UNK>'
    - Another new word is introduced in the word index– '<PAD>' to represent the words used to increase the length of shorter sentences to exactly 25

words. Populated embedding values with another set of random numbers (between -1 and 1) for the word '<PAD>'

The word index mapping is persisted so that same can be used to transform the data at the time of validating the results on test data sets.

- Spelling related feature creation – Following spelling related features are created and fed to CRF layer in DNN post feature encoding. All these features are created for each word of each sentence and 10 context words (5 next words and 5 previous words) within the sentence.
    - word.len(): number of characters in the word
    - word.istitle(): is the first letter of the word in upper case
    - word.isupper(): are all letters of the word in upper case
    - word.isdigit(): are all characters of the word digits
    - has_upper: helps to understand whether any letter in the middle of the word is upper case
    - has_number: helps to understand whether the word contains digits
    - has_symbol: helps to understand whether the word contain punctuation
    - word_format: all upper-case letter replaced by 'A', all lower-case letter replaced by 'a', all digits replaced by '0', all punctuations replaced by '-'. This the only spelling related feature that is not created for context words due to its high variability which had posed memory issue in the used infrastructure when one-hot encoding is applied.
    - word_format_summary - all consecutive upper-case letter replaced by 'A', all consecutive lower-case letter replaced by 'a', all consecutive digits replaced by '0', all consecutive punctuations replaced by '-',
    - BOS+n: Boolean value indicating the position of the word relative to the beginning of a sentence. 'n' indicates the relative position. E.g. BOS+1 will be True for only first word in the sentence and will be False for all other words.
    - EOS-m: Boolean value indicating the position of the word relative to the end of a sentence. 'm' indicates the position. E.g. EOS-1 will be True for only last word in the sentence and will be False for all other words.

See Figure 11 for how spelling related features are created for every word in a sentence. The same coloured values indicate that characteristics from context words are repeated as different features for surrounding words in the sentence. For example, features for ‘Word 1’ are also present as features of the previous word for ‘Word 2’ (second row, feature names starting with ‘-1’) and features for ‘Word 2’ are also present as features of next word for ‘Word 1’ (first row, feature names starting with ‘+1’).

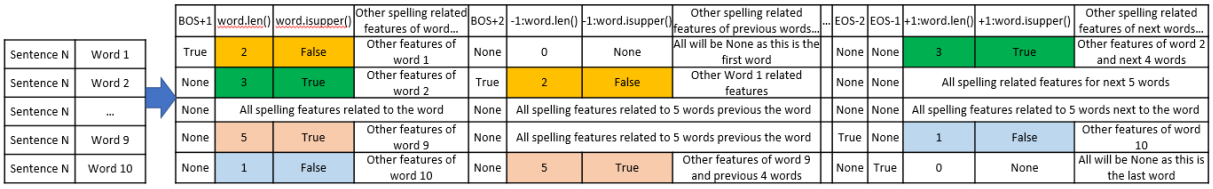

Figure 11: Spelling related feature creation for CRF

- Context related feature creation - POS and CHUNK attributes and its variations are used as context-related features.
    - postag: parts of speech tag as supplied in the training and testing dataset.
    - postag[:2]: Initial two characters of POS tag. This will indicate the root of the POS tag, e.g. if POS tag is ‘NNP’ (i.e. a proper noun) this feature will have value ‘NN’ (this indicates Noun)
    - chunktag: chunk or phrase information as supplied in the training and testing dataset.
    - chunktag[2:]: chunk or phrase tag excluding initial two characters. E.g. if the tag is ‘I-NP’ (i.e. intermediate word of a noun phrase), then this feature will have value as NP (i.e. noun phrase)

These features are also created for each word of each sentence and 10 context words (5 next words and 5 previous words) within the sentence. See Figure 12 for how context-related features are created for every word in a sentence. The same coloured values indicate that characteristics from context words are repeated as different features for surrounding words in the sentence. For example, features for ‘Word 1’ are also present as features of the previous word for ‘Word 2’ (second row, feature names starting with ‘-1’) and features for ‘Word 2’ are also present as features of next word for ‘Word 1’ (first row, feature names starting with ‘+1’).

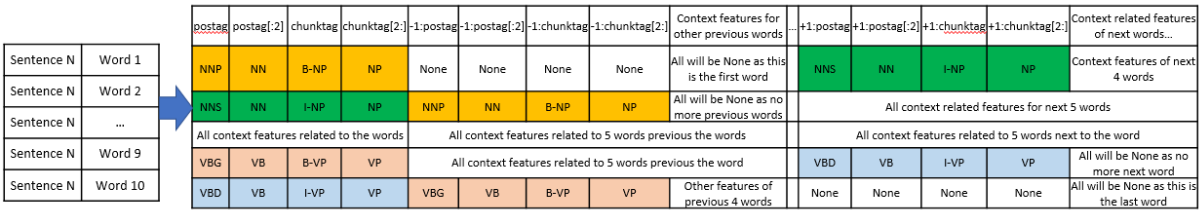

Figure 12: Context related feature creation for CRF

- Making CRF feature matrix uniform - As all sentences are not of the same length, all the spelling and context related features are not available for all words in the sentence. E.g. if a sentence has only 10 words, features related to 11 to 25th words (as decided number of words in a sentence is 25 words) will not be available while creating CRF features. See some examples of this in Figure 11 and Figure 12 where the feature value is 'None'. This needs treatment as DNN will expect a uniform numeric matrix as input. Therefore, for string and Boolean features, blank values are replaced by string 'NA' to indicate unavailability of the feature in that instance; for numeric features blank values are replaced by 0.

- One-hot encoding of spelling and context related features: One-hot encoding applied on following spelling and context related features to convert them to numbers. These features are of string or Boolean data type. Therefore one-hot encoding is applied onto all features except the features related to the number of characters in the words. This is applied to features for previous and next words as well.
    - spelling related features that are string –
        - word.istitle()
        - word.isupper()
        - word.isdigit()
        - has_upper
        - has_number
        - has_symbol
        - BOS+n
        - EOS-n
        - word_format
        - word_format_summary
    - all context-related features

- postag
- postag[:2]
- chunktag
- chunktag[2:]

The one-hot encoder and transformed feature names are persisted to be applied on the test dataset.

- Standardise numeric features – The spelling related feature for the number of characters in the word (applicable to context words too) is of numeric data type. Therefore, those features are standardised by removing the mean and scaling to unit variance. The mean and variances for each of the numeric features are retained to be applied on the test dataset.

4.3 NER model building using BI-LSTM-CRF

This section will discuss the architecture and hyperparameters used for NER model building. Figure 13 shows the high-level architecture with dimensions at each layer for the DNN model used for NER. Bi-LSTM-CRF model serves as the NER model that is later explained using post-hoc explainability techniques.

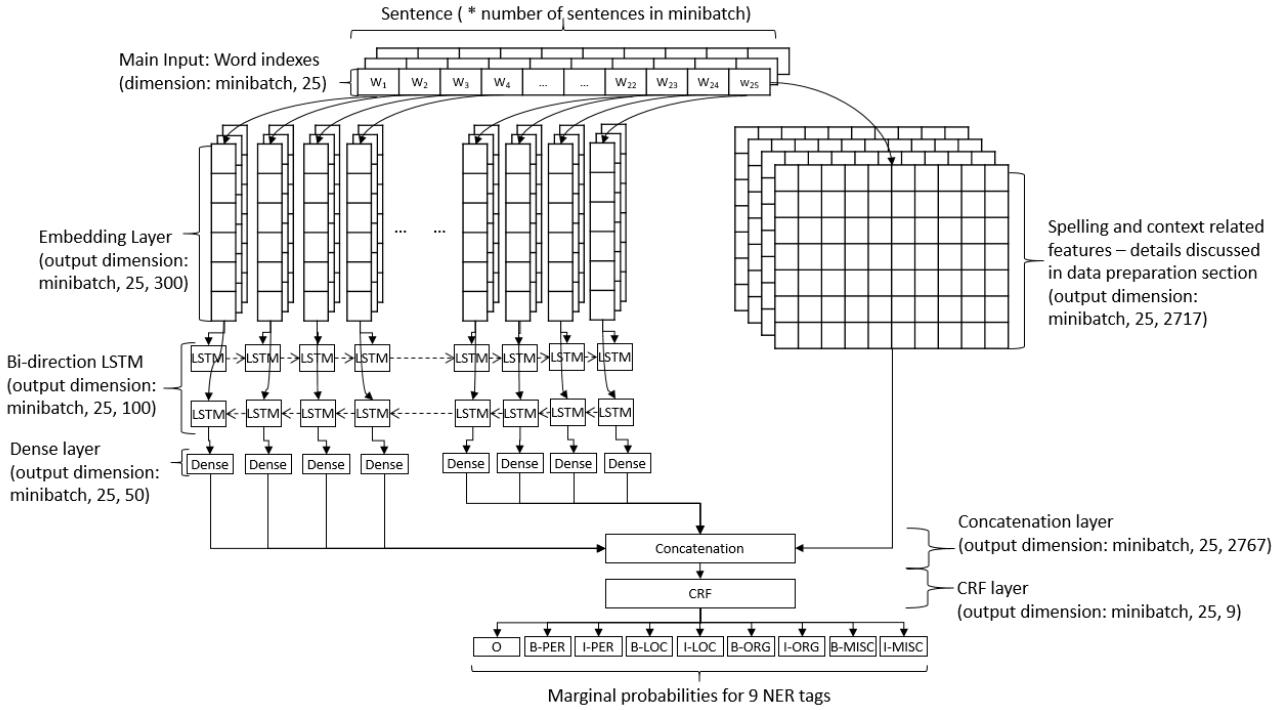

Figure 13: BI-LSTM-CRF architecture

4.3.1 Model architecture

Figure 13 shows the high-level architecture of the BI-LSTM-CRF model that is used for NER prediction. Table 9 provides further details of the architecture.

Table 9: BI-LSTM-CRF architectural details

Layer (type)	Input shape	Output shape	Units	Trainable parameters	Description
Input	Minibatch size x 25	Minibatch size x 25	NA	0	This accepts the word index for 25 words in a sentence
Embedding	Minibatch size x 25	Minibatch size x 25 x 300	NA	6419700	This converts each word to a 300-dimension vector as 300-dimension GloVe embedding vectors are used to initialize the weights.
Bi-directional LSTM	Minibatch size x 25 x 300	Minibatch size x 25 x 100	50	140400	As this is bidirectional LSTM layer to capture contextual features, outputs from forward pass and backward pass of the features get concatenated
Fully connected	Minibatch size x 25 x 100	Minibatch size x 25 x 50	50	5050	This is a fully connected layer.
Auxiliary Input	Minibatch size x 25 x 2714	Minibatch size x 25 x 2714	NA	0	This is an additional input containing spelling and context related features as discussed in section 4.2.2.
Concatenation	Minibatch size x 25 x 2714 AND Minibatch size x 25 x 50	Minibatch size x 25 x 2764	NA	0	The output of the dense layer and the auxiliary input will be concatenated for combined input to CRF layer.
CRF	Minibatch size x 25 x 2764	Minibatch size x 25 x 9	9	24984	This is the last and output layer of the model that outputs probability scores for entity tags for each word for each sentence.

4.3.2 Hyperparameters

Table 10 discusses the hyperparameters of the Bi-LSTM-CRF model. This table contains only the hyperparameters that are used in the experiment. There are many other available hyperparameters, that were not experimented with and thus not mentioned.

Table 10: Hyperparameter for BI-LSTM-CRF model

Layer	Hyperparameter	Value	Description
Embedding	Initialization weights	GloVe 300 embedding weights	Pre-trained GloVe embedding vector having 300 dimensions used to initialise the embedding layer weight which got trained as part of model training too.
LSTM	Kernel initialization weights	Truncated normal with mean 0 and standard deviation 0.1	Weights for LSTM is initialised using a statistical distribution or function of random normal distribution where a random number that is more than two standard deviations from the mean are discarded and redrawn form random numbers.
LSTM	Activation	Hyperbolic Tangent (tanh)	Activation function for candidate hidden state and output hidden state
LSTM	Recurrent activation	sigmoid	Activation function for input, forget, and output gates inside LSTM unit.
LSTM	Recurrent layer dropout	0.1	Dropout value used to regularize the flow of information across timesteps.
Fully connected	Activation function	Rectifier linear unit (ReLU)	Activation function for fully connected layer, ReLU is defined as maximum of 0 or input value {y = max(0, x)}
CRF	Learning mode	Marginal	Marginal learning mode is used to retrieve marginal probability distribution for each possible entity tag for a word. The other learning mode, join, cannot provide individual probability distribution.
NA	Optimizer	Adam	Optimization algorithm is chosen to find the optimum value for each of the trainable parameters based on the loss function. Adam is an adaptive learning rate optimization algorithm.
NA	Loss function	crf.loss_function	Loss function to be optimised by the model

NA	Evaluation metric	crf.accuracy F1 score	Performance metrics that are reported after each epoch for both training and validation splits
NA	Minibatch size	64	The number of sentences in each batch for training. These batches are shuffled after each epoch.
NA	Number of epochs	20	The number of epochs for training. The complete training data will pass through the DNN these many times.
NA	Validation split	0.1	The proportion of split for validation in training data while training in progress. This set of data is used to monitor the effect of training using the set of evaluation metrics selected.
NA	Reduce learning rate on plateau based on validation loss	patience = 3 factor = 0.2	Force reduction in learning rate after no reduction in validation set loss for 3 epochs (controlled by patience) and reduction factor is 0.2 (controlled by factor)
NA	Early stopping of training based on validation loss	patience = 5	Force termination of training if no reduction in validation set loss for 5 epochs (controlled by patience)

4.3.3 Model evaluation

Test-a dataset is used for validating the Bi-LSTM-CRF model. Same training data preparation steps are followed except three steps highlighted in Figure 14. These three steps are discussed below.

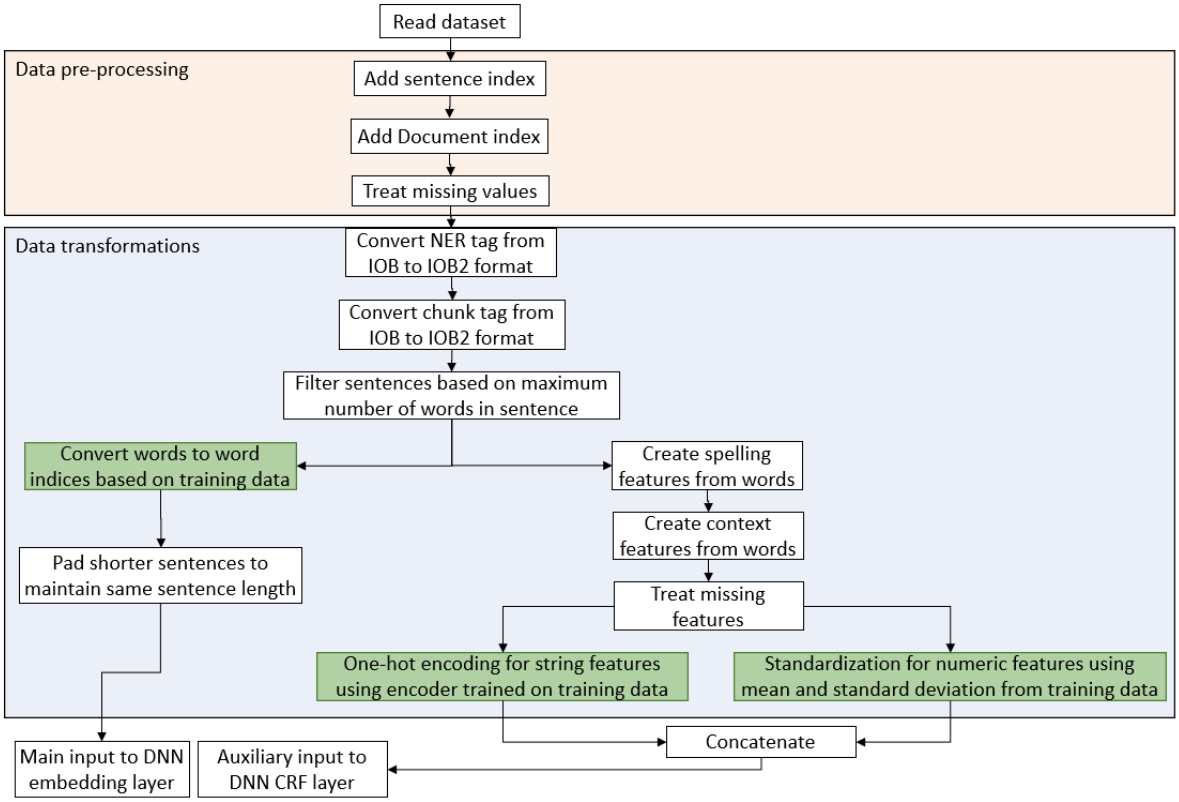

Figure 14: Data preparation steps for test data

- Word to word index transformation – converting word to word indices are done based on the word index assignment at the training data preparation. I.e. any new word in the test-a dataset is mapped to the word index of <UNK> irrespective of their number of occurrences in the test-a dataset.
- One-hot encoding of textual spelling and context related features – this is done using one-hot encoder persisted at the time of training data preparation.
- Standardisation of numeric spelling related feature – Mean and standard deviation values learned for training data numeric features are used to apply this step on test data.

The performance of the NER model is evaluated using the sequence F1 score. Like F1 score, Sequence F1 is the harmonic mean of recall and precision, where precision for an entity is calculated as the proportion of the number of accurate prediction over the

number of total predictions for that entity and recall for an entity is calculated as the number of accurate prediction over the total number of ground truth instances for that entity. In these cases, the entity is defined by the combination of the entity tag (e.g. LOC, MISC etc. derived from the sequence of B tag and I tags), and start and end position of the tag in the sentence (e.g. position 0 to 2 in the sentence).

4.4 Explaining the NER model using LIME-NER and Kernel SHAP

This study has used LIME that is improvised for NER by (Villarroya, 2018) and Kernel SHAP as two locally interpretable post-hoc techniques for explaining the NER model. This section will discuss the steps and variations followed in generating the explanations. While the high-level steps are similar in nature for both the explanation techniques, details for the steps in generating the explanation as shown in the 'Generate explanation' block in Figure 15 are significantly different.

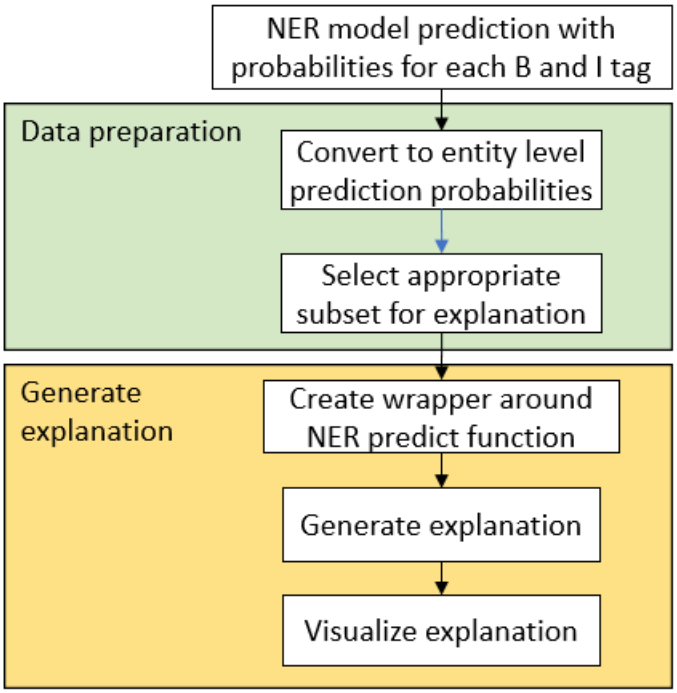

Figure 15: High-level process for generating explanations

4.4.1 Data preparation for explanation

As none of the explanation techniques is built to support sequence tagging tasks, few steps need to be performed before NER predictions can be used for generating explanations.

- Test-a dataset predictions for explanation – This study has used the NER model built as per the steps described in section 4.2 and 4.3 to generate NER predictions on the test-a dataset. The steps described in section 0 are used to generate the prediction probabilities. As the model is trained on IOB2 tagging scheme, the generated predictions are probability vector of length 9 (O, B-PER, I-PER, B-LOC, I-LOC, B-ORG, I-ORG, B-MISC, I-MISC) for each word in every sentence.
- Conversion to entity level prediction probabilities – This step is required to facilitate selection of predicted instances for explanation based on prediction probability score for entities. As suggested in (Villarroya, 2018), this study converted the probability vector containing probabilities for all B and I tags to a probability vector having aggregated probabilities for each entity. I.e. each word has a probability vector of size 5, having the probability values for O, PER, ORG, LOC, and MISC. The aggregation of probabilities from B and I tag to entity level probability is done as per the following logic.
    - For a word, if 'O' tag probability has the highest value then 'O' tag original probability is retained for 'O' entity tag. For the same word, maximum of B and I tag probability for an entity is copied as entity tag probability.
    - For a word, if a B or I tag of an entity has the highest probability value, that gets averaged over all the subsequent words until either 'O' tag or another B tag (including B tag of same entity type) probability is maximum for a word. For the same set of words, maximum of B and I tag probabilities are averaged for other entity tags as well. For the same set of words 'O' tag probabilities are averaged as well. Post this aggregation, for all words in this set, entity level probability scores are the same.
- Selection of instances for the explanation – generation of local explanation using either LIME-NER or Kernel SHAP is both compute and memory expensive. Therefore, the study has used a subset of predicted instances for explanation. The selection is done based on multiple factors in different scenarios as shown in Table 11. Details of the subset selection are covered while discussing relevant scenarios.

Table 11: Explanation instance selection levers

Selection Criteria	Description
Entity	All explanations are generated for either 'LOC' or 'MISC' entities as 'LOC' is the best performing entity and 'MISC' is worst performing entity as per the NER model performance metrics on the test-a dataset.
Entity level probability threshold	A threshold of 0.95 is used irrespective of accurate and inaccurate prediction in most of the scenarios with the exception in the evaluation of explanation that is discussed later.
Multiword entities	It is observed that explanation of multiword entities is more difficult than single word entities. Therefore, this filter criterion is used in several scenarios especially for explaining accurate predictions. However, there are few scenarios where this criterion is relaxed to increase the number of instances for explanation.
Inaccurate prediction	This subset is used for qualitative and quantitative evaluation metrics calculations.
Unknown entity words	Unknown entity words refer to words that are part of the predicted entity but are not part of training vocabulary (i.e. word index is not assigned to these words; if encountered at the time of prediction, word index of word '<UNK>' is assigned). Explanation for these cases is expected to highlight the influence of context words which is subject of interest.
Random selection	This subset selection is done to reduce the number of instances for explanation after applying multiple of above-mentioned filters.

4.4.2 LIME-NER Explanation

Original LIME, as proposed by (Ribeiro et al., 2016), has a capability for explaining text where predicted output expected to be a single probability vector, e.g. text classification. In the case of NER prediction, the prediction output is multiple probability vectors, one for each word in the sentence. Therefore, a wrapper is applied to the NER prediction function to make the problem suitable for LIME text explainer.

- NER prediction wrapper – LIME text explainer expects a prediction function that can be applied on a text to get the probability vector for target classes. The NER prediction wrapper achieves this in three steps based on the assumption that explanation is sought for a single entity (that can comprise of multiple consecutive words) in a sentence.
  - Pre-processor – LIME text explainer presents the original sentence and perturbed sentences to the prediction function to retrieve the probability score vector against each of the sentences. Therefore, it is required to pre-process and transform the original and perturbed sentences so that it can be used by the trained NER model for prediction. This pre-processing step performs a part of the data transformation steps that are discussed in section 0 to make the sentences compatible with NER prediction. Figure 16 shows the steps that this pre-processor performs.

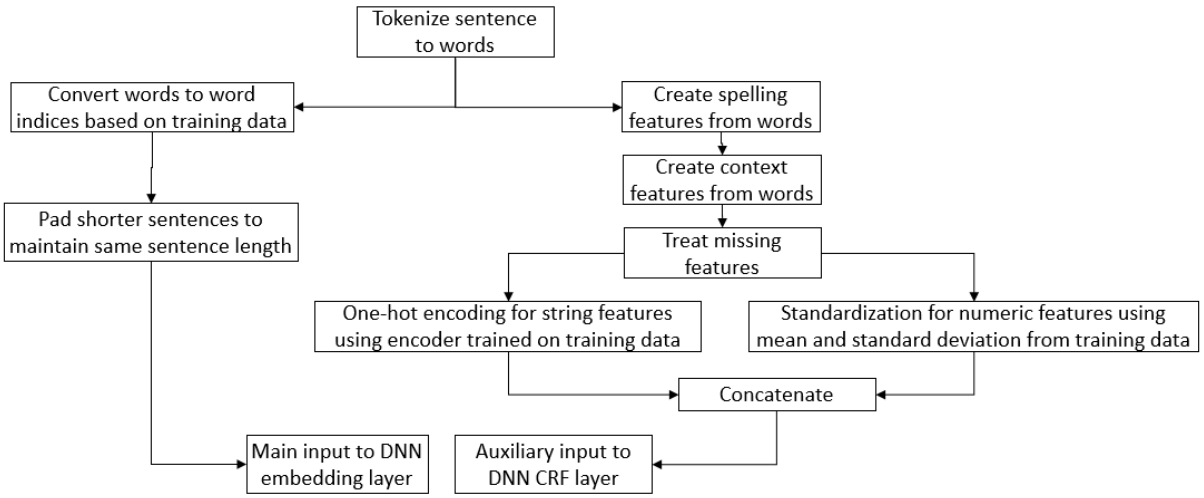

Figure 16: Pre-processing steps in NER prediction wrapper

It is worth noting that LIME internally uses dummy words to replace original words in sentences to generate perturbed sentences. These dummy words are represented with the word index of '<UNK>' (as defined in section 4.2.2) when provided as input to the NER model. The spelling and context related features

for such dummy words are populated with fixed values that indicate unavailability of the feature, i.e. string and Boolean features are populated with string ‘NA’ and numeric features are populated with 0.

- Prediction – The transformed features for original and perturbed sentences are fed to trained NER model to generate the probability vector for each word in the sentence. However, LIME expects a single probability vector for each sentence. Therefore, these probability vectors are post processed so that a single vector can be returned.
- Post-processor - As discussed in (Villarroya, 2018), generation of NER explanation in entity level needs aggregation of output probability vector (of length nine) to entity level probability vector (of length five). It is also required to process the probability vectors for each word to result in a single probability vector for the sentence. The post-processor achieves these two objectives by selecting only the probability vectors for the set of words corresponding to the entity tag that needs explanation. Then post-processor aggregates the B and I tag probabilities for those set of words using following rules – (1) for each entity tag, average over B tag probability for the first word and I tag probabilities for all subsequent words, (2) for O tag, average over O tag probabilities for all the words in the set. This results in a reduced length probability vector representing entity level probabilities for each word. Probabilities are now the same for each of the words in the set. Postprocessor returns one such probability vector for each of the sentences for LIME to build the locally interpretable model.

- Generate explanation – Parameters in Table 12 are used to set up LIME text explainer for NER model.

Table 12: LIME parameters

Parameter	Description
bow	This is a Boolean flag that indicates whether to consider the word tokens as a bag of words or position of the words in the sentence carries any significance. This is set to false for the study as NER is sequence modelling task and position of each word has an impact on the predicted tag for surrounding words.
split_expression	Space is used as the expression for splitting sentences in words as the training and testing data sets have words split in the same way.
num_samples	The number of perturbed sentences that will be generated by LIME. The default value of 5000 is used in all experiments.
Number of features in explanation	This decides sparsity of locally interpretable model. The number mentioned for the parameter decides on the number features used in the local model and same is used to visualise the explanations. Different values are used for this parameter in different scenarios. E.g. for visualisation top 5 features for each entity is displayed and for evaluation metrics calculation top 10 features are used.
Labels	This parameter is used in a multiclass prediction setup to tell LIME the target class for explanation. Another related parameter is used to generate explanation for two competing classes at the same time in a few select scenarios.
Entity	As per (Villarroya, 2018), to use LIME for NER prediction explanation, the perturbation methodology inside LIME needs a change. Original LIME implementation was updated so that words that are part of an entity is perturbed together, but not individually. To use that implementation, indexes of entity words are provided in the entity parameter while setting up LIME for NER explanation.

- Visualise explanation – LIME supports visualisation of explanation using two different visuals. Figure 17 presents explanations generated from LIME as feature weights from the locally interpretable model for each of the words. Figure 18 presents the same explanation via highlighted words. Darker colour indicates higher feature importance.

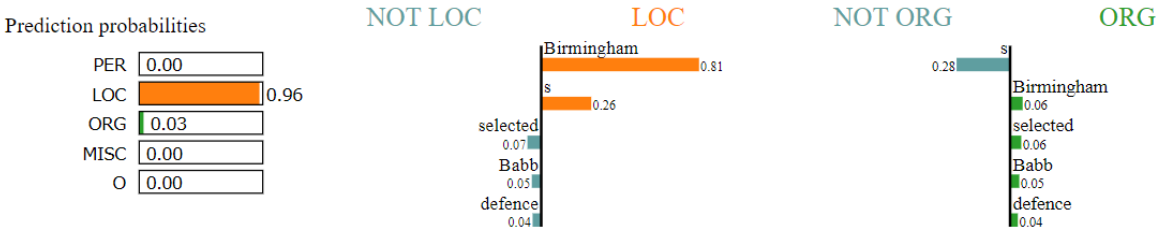

Figure 17: Variable importance visualisation in LIME

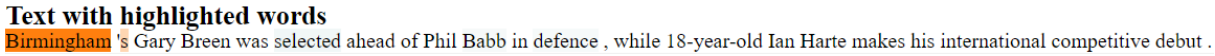

Figure 18: Text with highlighted important words in LIME

4.4.3 Kernel SHAP for NER Explanation

Kernel SHAP (Lundberg and Lee, 2017) does not accept text input by design. It expects two-dimensional data as input for explanation where one dimension is the number of samples and another is the number of features. Similar to LIME, Kernel SHAP expects predicted output to be a single probability vector. With respect to the NER model, neither input nor output is in the format of that Kernel SHAP expects. Therefore, an additional data preparation step is created to shape the input and a prediction wrapper is created to transform NER prediction output as required. Additionally, Kernel SHAP needs feature names for presenting the explanations. This feature list is created and enriched separately and discussed later in the section.

- Flatten NER input – Kernel SHAP expects inputs to be provided as a feature vector so that it can create coalition vectors that are used to understand the importance of feature coalitions. Sentences, that needs explanation, are processed to create the flat feature vector. Activities in this step are very similar to the pre-processor in the NER prediction wrapper created for LIME explanation. However, few activities are different which are highlighted in Figure 19. The highlighted steps include flattening spelling and context related features for every word.

Originally, these features are created for every word in a sentence. Therefore, the dimension of these features is (number of words in a sentence X number of features). This got flattened to create an array of length (number of words in a sentence * number of features) for each sentence. This array is then concatenated with the array of word indices corresponding to the words in the sentence. This becomes an array of length (number of words in a sentence + number of words in a sentence * number of spelling and context features). In this study, the length was 67,875, i.e. equal to 25 (number of words in the sentence) + 25 * 2,714 (number of encoded spelling and context related features).

The details of other steps that lead to the creation of main and auxiliary input to DNN are already discussed in section 0 as part of data transformation steps required for NER model validation. Main and auxiliary input then gets flattened using the method mentioned above. While the data transformation steps are part of pre-processor of NER prediction wrapper for LIME explanation (section 0), Kernel SHAP implementation need them to be outside of the NER prediction wrapper due to difference in accepted input type for these two methods.

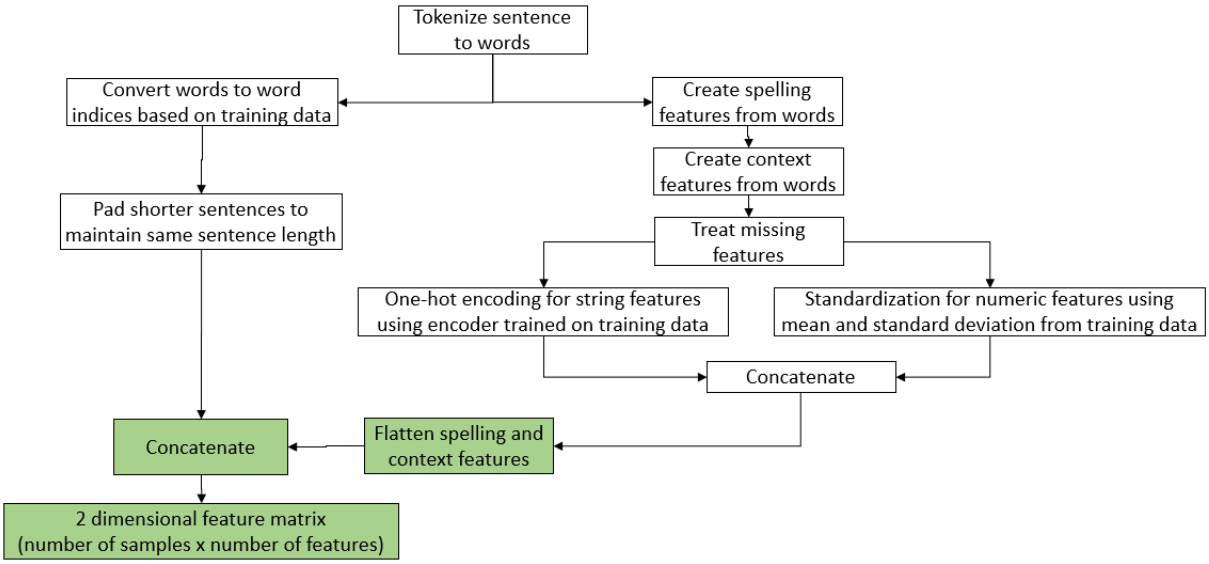

Figure 19: Flatten input for Kernel SHAP

- Enrich feature list – Kernel SHAP explanations are presented using the feature names of the flattened input list. In the case of NER model, there are two types of features, the words and the spelling and context related features. For a better understanding of explanations, this feature list is enriched in the following ways.

- Word features – In the flattened input list first 25 elements are populated with word indices corresponding to the sentence. As the word position alone is not enough to express any message, the word index is first converted to the actual word in the sentence using word index to word mapping persisted from training. Then the parts of speech tag from the input data set, the position of the word in the sentence and predicted NER entity tag are added in the feature names, e.g. U.S.>5>NNP>LOC. Figure 20 shows the display format for word features. There is a special case where the original word is not present in the word to word index mapping learned at the time of training. These words appear as 'word|<UNK>' to indicate that the word is out of training vocabulary, e.g. 'Ariel|<UNK>'. Other details (POS, position in the sentence and predicted entity) of the word will be added as usual.

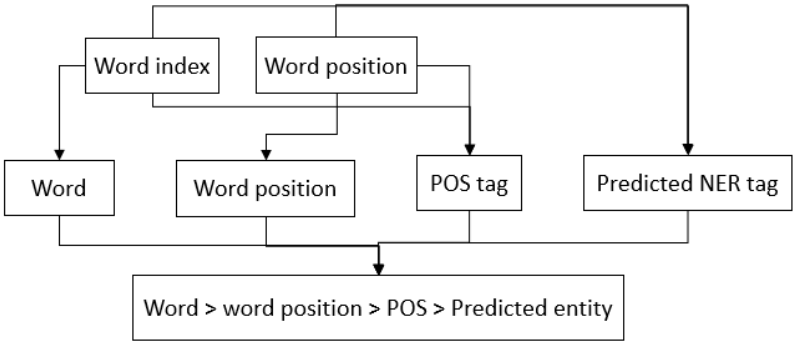

Figure 20: word feature format

- Spelling and context related features – The feature names for spelling and context related features are meaningful to some extent (see feature names in section 4.2.2, Figure 11 and Figure 12). However, the feature names do not contain the word for which the feature value is derived. The same feature appears for a word and the words surrounding the word. Therefore, these features are enriched in the following two ways.
    - If the feature value is for a word then the feature will look like: <Word>__<feature name>, e.g. took__word.len(), this is number of characters of word 'took'
    - If the feature value is for a surrounding word then the feature will look like <Word>:<relative position>:<surrounding word>__<feature name>, e.g. took:+3:points__word.len(), this is the number of characters of word 'points' which appeared three words after ('+' indicates after) the word 'took'.

- NER prediction wrapper – As discussed in the previous step, sentences will be converted to flattened input features before feeding to Kernel SHAP for explanation. However, it is also required to convert them back to NER model compatible input format when Kernel SHAP presents them for NER model prediction. The output probability vector list for a sentence from NER model prediction needs transformation as Kernel SHAP expects a single probability vector per array of flat feature list (representing one sentence). Like LIME, the NER prediction wrapper achieves this in three steps based on the assumption that explanation is sought for a single entity (that can comprise of multiple consecutive words) in a sentence.
    - Pre-processor – When Kernel SHAP presents the flat input array for a sentence for NER prediction, that needs to be converted back to the input format understood by trained NER model. This is done based on the number of words in the sentence and the number of encoded spelling and context related features that were created at the time of training for each word. Once the flat feature input is split into the array of word indices and matrix of encoded spelling and context related features, that can be fed into the NER model for prediction. Flattening of the input feature (that is discussed earlier in this section) is performed only on the original sentence that is selected for explanation. However, the conversion from flat input feature to NER model compatible format is performed on multiple perturbed feature list that is created by Kernel SHAP.
    - Prediction – The transformed features for original and perturbed feature list are fed to trained NER model to generate the probability vector for each word in the sentence. These probability vectors are processed so that a single vector can be returned.
    - Post-processor – Functionality of post-processor in NER prediction wrapper for Kernel SHAP is the same as the post-processor in NER prediction wrapper for LIME NER that is discussed in section 0.

- Generate explanation – Parameters in Table 13 are used to set up Kernel SHAP explainer for NER model.

Table 13: Kernel SHAP parameters

Parameter	Description
Reference data	Kernel SHAP requires reference data for creating coalition vector, i.e. replacing values for a set of features to understand the importance of other features on prediction or importance of the absence of these features on prediction. For this experiment, a reference sentence is created by repeating '<UNK>' 25 times (i.e. maximum number of words in a sentence for this study). Then that is processed with the flatten input list step to create a flat feature list. This results in an array of length 67,875. First 25 places in the array were filled with the word index for '<UNK>' word that is defined in section 4.2.2. Remaining places in the array representing spelling and context related features are filled with 0 indicating unavailability of the feature.
link	A generalized linear model link that connects feature importance values to the locally interpretable model output. The default value of 'logit' is used for all experiments that converts feature importance values (i.e. SHAP values) to log-odds of probabilities
l1_reg	This represents the feature selection method for the locally interpretable model. The number of features that are used to finally explain the prediction is dependent on this parameter. The possible values are aic (Akaike information criterion), bic (Bayesian Information Criteria), and num_features(int). AIC and BIC are both penalized likelihood criteria used to select the right set of features for a model. As penalty in BIC is dependent on the number of samples, it has higher chances of selecting a smaller (less complex) model compared to AIC. (AIC vs. BIC – The Methodology Center, 2020) Therefore, the study used bic as the l1 regularisation parameter for all the experiments.

nsamples	This decides the number of perturbed samples that are generated and presented to the NER model for prediction. The experiments used default value for this parameter which means the number of such samples is (2 * Number of features + 2048). However, as the study used a single reference data, if the instance for explanation and reference data has the same value for a feature, then that feature is excluded from the effective feature set for 'nsamples' calculation. Having the same value in an instance of explanation and reference data does not allow the creation of a coalition for that feature.

- Visualise explanation – Kernel SHAP explanation generates a list of length same as the number of entities, i.e. five for this study. Each item in this list is an array of dimension (number of samples sent for explanation X number of features). As the experiments are trying to explain individual NER predictions separately, each item in the list has a dimension of (1 X 67,875), This study uses four different types of plots provided by SHAP framework to explain single NER prediction.
    - Force plot – This plot shows how different features increase or decrease the predicted probability for an entity. Base probability value for the entity is indicated on the plot as well. Base probability is the NER prediction probability for the target entity on the reference data. Blue colour indicates the features that decreased the probability and the red colour indicates the features that increased the probability. Figure 21 shows feature influence for prediction as 'PER' entity for the words 'Anne-Gaelle Sidot' in the sentence '8 - Lindsay Davenport (U.S.) beat Anne-Gaelle Sidot (France) 6-0 6-3'

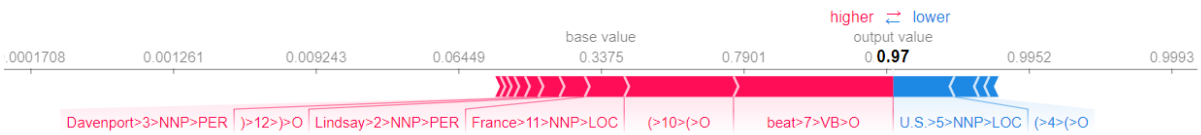

Figure 21: Kernel SHAP - force plot

    - Multiple output decision plot – Decision plot explains feature importance for the model's multiple outputs for a single observation. In this case, the x-axis represents the raw SHAP values in log odds of probabilities. Different coloured lines indicate different entities as shown in the legend. The straight vertical line indicates average base SHAP value for all entities. The plot shows how each of

the impacting features changes the model output for each entity. It is very evident from Figure 22, model's prediction of entity of 'PER' is well supported by explanation as the SHAP values for other entities are insignificant compared to SHAP value for 'PER' entity.

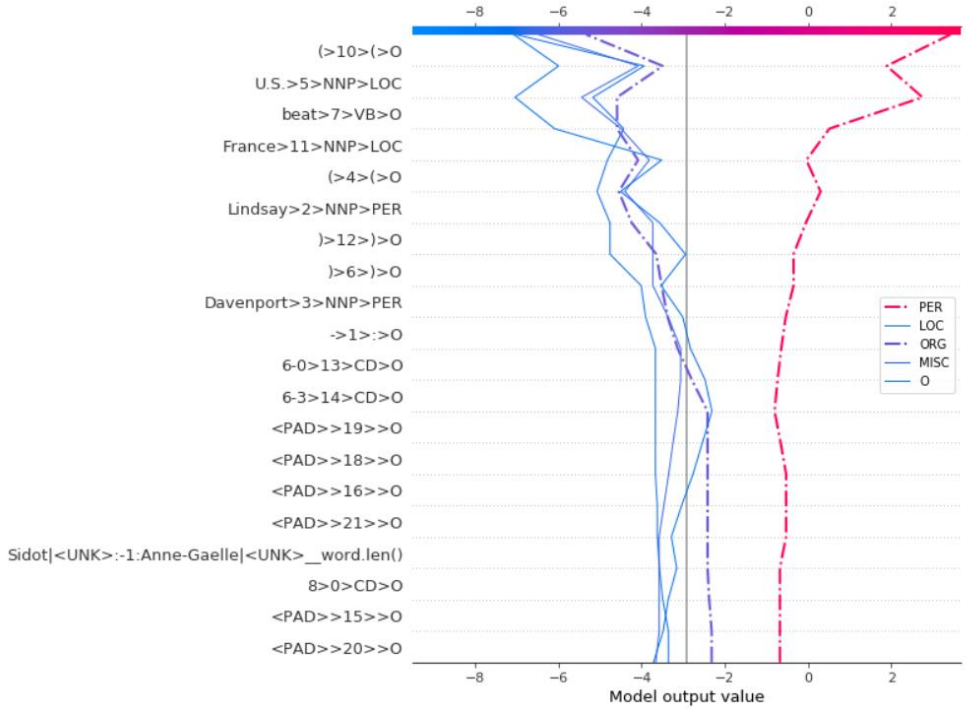

Figure 22: Kernel SHAP - Multioutput decision plot

- Summary plot for multiple entities together - summary plot shows mean SHAP values (in log odds of probabilities) for each entity if provided with a list of SHAP values as retrieved from Kernel SHAP explainer. This plot can highlight the most important feature for prediction of any of the entities. E.g. Figure 23 indicates '(', that comes just after the word 'Anne-Gaelle Sidot', has the highest influence on prediction for the words 'Anne-Gaelle Sidot' in the sentence '8 - Lindsay Davenport (U.S.) beat Anne-Gaelle Sidot (France) 6-0 6-3'

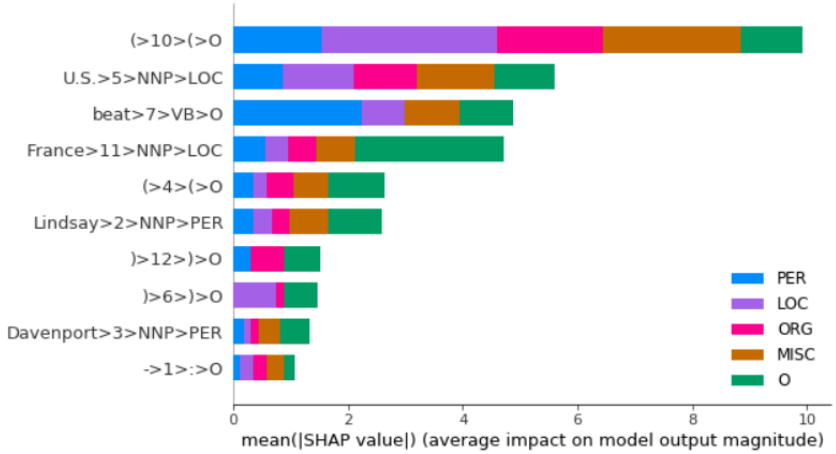

Figure 23: Kernel SHAP - Summary plot - multiple entities

- Summary plot for a single entity – when SHAP values for a single entity are provided to the summary plot, it shows scatter plot of SHAP values for each feature to indicate the impact of each feature on the predicted probability score for that entity. E.g. Figure 24 indicates 'beat' has the highest positive influence on prediction as 'PER' entity for the words 'Anne-Gaelle Sidot' in the sentence '8 - Lindsay Davenport (U.S.) beat Anne-Gaelle Sidot (France) 6-0 6-3'. This shows the negative contribution of features by plotting the negative SHAP values as shown for the word 'US'. The features are placed in descending order of absolute value of SHAP values, i.e. the feature having highest absolute SHAP value is placed at the top of the plot.

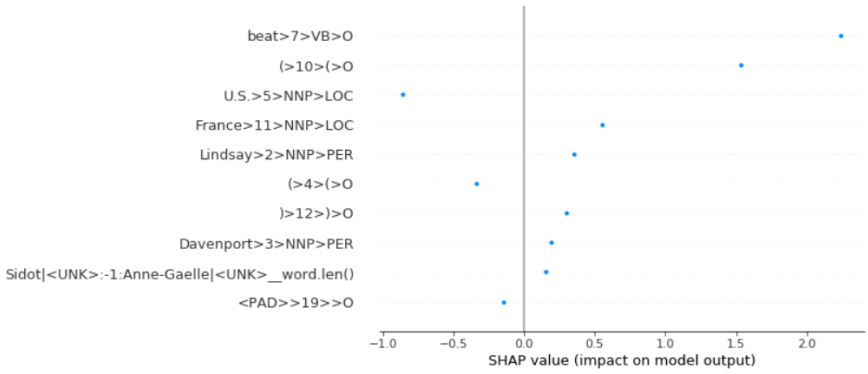

Figure 24: Kernel SHAP - single entity summary plot

4.5 Evaluation of NER explanations

This section will discuss the steps performed to evaluate the explanation generated form Kernel SHAP and LIME-NER. All different type of evaluation needs different type of data preparations and metrics calculation that will be discussed in this subsection.

4.5.1 Qualitative evaluation

The study has implemented three different qualitative evaluation methods to compare LIME-NER and Kernel SHAP explanation with respect to the accuracy, time efficiency and comprehensibility.

- Accuracy – To compare the accuracy of explanation, the study focused on instances where the prediction of an entity from the NER model is accurate and entity prediction probability for two entities are similar. Visualizing explanation for those instances is expected to answer why the finally predicted entity is chosen over the other entity for which the NER prediction probability is very close. The selection of these instances is the two-step process.

  In the first step, the study selects the instances based on rules in Table 14.

Table 14: Explanation instance selection for evaluating the accuracy

Selection Criteria	Values for only accurate explanations
Entity	MISC and LOC
Entity level probability threshold	0 i.e. no threshold
Multiword entities	Both single and multiword entities
Random selection	Random selection is done after the second step

  In the second step, the following are computed for LOC and MISC entities individually.

  - Selected the instances where the entity has the highest prediction probability, i.e. the final predicted entity
  - Then calculated standard deviation for the NER model predicted probabilities for that entity in those instances.
  - Chosen only those instances where the prediction probability for another entity is within one standard deviation of prediction probability of predicted entity.

  Selected 10 random instances from for each MISC and LOC separately to generate and visualise explanations using LIME-NER as per the steps discussed in section 0 and using Kernel SHAP as per steps discussed in section 4.4.3

- Time efficiency – Time efficiency of explanation is calculated based on the number of elements used to explain a prediction instance. Explanation instances are selected using the rules in Table 15 for this metrics calculation.

Table 15: Explanation instance selection for time efficiency metrics

Selection Criteria	Values for accurate explanations	Values for inaccurate explanations
Entity	LOC and MISC	LOC and MISC
Entity level probability threshold	0.95	0.95
Multiword entities	Multiword only	Both single and multiword entities
Random selection	10 random samples for LOC and 10 random samples for MISC	All of 7 samples for LOC and all of 2 samples for MISC.

The explanation instances selected are then explained using LIME-NER as per the steps discussed in section 0 and using Kernel SHAP as per steps discussed in section 4.4.3. In this case, the number of features is not restricted for LIME-NER, i.e. LIME explanations used as many as feature required for an accurate explanation.

As discussed in section 3.3.2, not all elements used in explanation are equally meaningful for explaining a NER prediction. Therefore, a weighting scheme is used for different type of features that appear in the explanations. The weighting scheme used is provided in Table 16. Notably, LIME-NER presents its explanation only using original word features, thus the explanation feature weights for LIME NER explanation is always 1 for each word. The explanation provided by Kernel SHAP has all types of features (the original word, spelling and context related features); weights will be assigned based on the type of feature selected for explanation.

Table 16: Explanation feature weights for time efficiency calculation

Feature type	Time efficiency weight	Reason for this weight
Original word features	1	The word is the basic unit of explanation, so it is assigned unit weight.
Number of characters in the word	1.1	It is not apparent how the number of characters can facilitate explanation; thus, higher weight is assigned.

Are all letters of the word in upper case	0.8	This feature is likely to highlight word format pattern where this word pattern is easy to understand, thus a lesser weight is assigned.
Is the first letter of the word in upper case	0.8	This feature is likely to highlight word format pattern where this word pattern is easy to understand, thus a lesser weight is assigned.
Are all characters of the word digits	0.8	This feature is likely to highlight word format pattern where this word pattern is easy to understand, thus a lesser weight is assigned.
Whether any letter in the middle of the word is upper case	0.8	This feature is likely to highlight word format pattern where this word pattern is easy to understand, thus a lesser weight is assigned
Whether the word contains both letters and digits	0.8	This feature is likely to highlight word format pattern where this word pattern is easy to understand, thus a lesser weight is assigned
Does the word contain punctuation	0.9	This feature is likely to highlight word format pattern where this word pattern is easy to understand, thus a lesser weight is assigned
Word format	1.2	This is related to word morphology, but encode a lot of patterns together which is not easy to understand. Thus, a higher weight is assigned.
Word format summary	1.2	This is related to word morphology, but encode a lot of patterns together which is not easy to understand. Thus, a higher weight is assigned.
Position of the word relative to the beginning of the sentence	1.5	This feature does not seem to be intuitive in explaining prediction. Thus, a higher weight is assigned.
Position of the word relative to the end of the sentence	1.5	This feature does not seem to be intuitive in explaining prediction. Thus, a higher weight is assigned.
Parts of speech tag	0.7	Parts of speech is easily understood having language understanding, thus assigned a lesser weight.

Chunk / phrase tag	1.2	Chunk/phrase tag can be understood by a person having depth in language understanding. Thus a higher weight is assigned

For each of the combinations in Table 17, explanation is generated using LIME-NER as per the steps discussed in section 0 and using Kernel SHAP as per steps discussed in section 4.4.3. Using the features from the explanation and feature weights from Table 16, time efficiency is calculated as per the formulation of time efficiency metrics in Eq(1).

Table 17: Time efficiency calculation combinations

Sl.No	Explanation method	Entity	Accurate / Inaccurate
1.	LIME-NER	MISC	Accurate
2.	Kernel SHAP	MISC	Accurate
3.	LIME-NER	MISC	Inaccurate
4.	Kernel SHAP	MISC	Inaccurate
5.	LIME-NER	LOC	Accurate
6.	Kernel SHAP	LOC	Accurate
7.	LIME-NER	LOC	Inaccurate
8.	Kernel SHAP	LOC	Inaccurate

- Comprehensibility – comprehensibility of explanation is understood based on the similarity and dissimilarity of words used for explanation. A T-distributed Stochastic Neighbour Embedding (t-SNE) model is built using the output of the embedding layer (i.e. first hidden layer of BI-LSTM-CRF NER model). The output of this layer represents every word as a 300 dimension vector, that got initialised by the pre-trained GloVe embedding wights and the weights are modified as part of the NER training. The t-SNE model transforms these 300-dimension vectors to 2-dimension vectors for easy visualisation.

  t-SNE model is built using all the words that got selected for the word to word index assignment in the training stage. Target dimension is set to 2 and PCA initialisation used for the t-SNE model building.

  Instances for explanation are selected using rules mentioned in Table 18 for visualisation of word similarities and dissimilarities.

Table 18: Explanation instance selection for comprehensibility visualisation

Selection Criteria	Values for accurate explanations
Entity	LOC and MISC
Entity level probability threshold	0.95
Multiword entities	Multiword only
Random selection	35 random samples for LOC and 35 random samples for MISC

Explanations are generated using LIME-NER as per the steps discussed in section 0 and using Kernel SHAP as per steps discussed in section 4.4.3.

Top 5 word-features are selected from LIME-NER and Kernel SHAP explanations for each of the combinations in Table 19. Non-word features (e.g. number of characters in word) in Kernel SHAP explanation are excluded for this visualisation.

Table 19: Word similarity visualisation scenarios

Sl.No	Entity	Explanation technique	To evaluate
1.	MISC	LIME-NER vs Kernel SHAP	Identity & Stability
2.	LOC	LIME-NER vs Kernel SHAP	Identity & Stability
3.	LOC vs MISC	LIME-NER	Separability
4.	LOC vs MISC	Kernel SHAP	Separability

t-SNE embedding for all words from LIME-NER and Kernel SHAP explanations are visualised together for the first two scenarios in Table 19. Words that are common in both explanation, words that only appeared in LIME-NER explanation and words that only appeared in Kernel SHAP explanation got painted in different colours for analysing the similarity of words. These visualization helps understand whether explanations generated from two different methods for the same entity predictions are using similar words, if not same.

t-SNE embedding for the top 5 words from each of 'LOC' and 'MISC' entity prediction explanation using one method is visualised together for the next two scenarios in Table 19. Words that are common in both entities, words that only appeared in 'LOC' entity prediction explanation and words that only appeared in 'MISC' entity prediction explanation got painted in different colours for analysing dissimilarity of words. These

visualization helps understand whether explanations generated from two methods for different entity predictions are using different words.

4.5.2 Quantitative evaluation

The study uses AOPC as the quantitative metrics for comparing Kernel SHAP and LIME-NER explanations.

The area over the perturbation curve (AOPC) is a perturbation based quantitative metric that is used by (Villarroya, 2018) for LIME-NER implementation. This study has implemented AOPC for Kernel SHAP and compared results against LIME-NER explanation. Instances for explanation for this metrics calculation are selected using rules mentioned in Table 20.

Table 20: Explanation instance selection for AOPC metric

Selection Criteria	Values for accurate explanations	Values for inaccurate explanations
Entity	LOC and MISC	LOC and MISC
Entity level probability threshold	0.6	0.6
Multiword entities	Multiword only	Both single and multiword entities
Random selection	10 random samples for LOC and 10 random samples for MISC	10 random samples where ground truth class is LOC but predicted as a different entity and 10 random samples where ground truth class is MISC but prediction as a different entity

AOPC with MoRF metrics calculation needs perturbation of explanation feature list in the order of importance for ground truth entity class to understand how much the explanation agrees with the model prediction. This is useful while evaluating explanations for correctly predicted instances. Similarly, AOPC with LeRF metrics calculation needs perturbation of explanation feature list in reversed order of importance for the ground truth entity class. For incorrect predictions, removing the least important features with respect to explanation generated for ground truth class is expected to increase the prediction probability of ground-truth class. The relevance of AOPC with MoRF and LeRF is discussed in Table 5.

Both MoRF and LeRF methods need explanation method to return the list of influencing features in order of importance. Therefore, post generation of explanation for ground truth class using LIME-NER as per the steps discussed in section 0 and using Kernel SHAP as per steps discussed in section 4.4.3, feature lists with non-zero scores are retrieved in the order of scores assigned by the respective methods.

In the next step, the features are perturbed in the original sentences to calculate the impact on ground truth entity probability. However, the steps of perturbation are different for LIME-NER and Kernel SHAP.

In the case of LIME-NER, the features as extracted from explanation are words from the original sentence. Therefore, perturbation is performed by replacing words by the word '<UNK>' in order of importance (order of most importance for MoRF and order of least importance for LeRF) in each iteration. Post perturbation, the predicted probability score is recorded for the ground truth entity. The steps discussed in section 0 for NER prediction wrapper in the context of LIME-NER are used to retrieve the entity prediction probability for the ground truth entity.

In the case of Kernel SHAP, the list of features is enriched word features and enriched spelling and context related features. The nature of the feature is understood before performing the perturbation. The word features are replaced by the word index corresponding to the word '<UNK>'. Encoded spelling and context related feature values are toggled (i.e. 1 to 0 and 0 to 1) if they are of the Boolean data type. The numeric spelling related features set to 0 values. Post perturbation, the predicted probability score is recorded for the ground truth entity. The steps discussed in section 4.4.3 for NER prediction wrapper in the context of Kernel SHAP are used to retrieve the entity prediction probability for the ground truth entity.

AOPC is derived with the MoRF method as per Eq(3) and with LeRF method as per Eq(5) for predicted probability lists generated for each of the combinations in Table 21. There were 15 iterations of perturbations (one feature removed/replaced in each iteration) for each of the selected sentences for both AOPC methods and both explanation methods.

Table 21: AOPC combinations

Sl.No	Explanation method	Entity	Accurate / Inaccurate	AOPC method
1.	LIME-NER	MISC	Accurate	MoRF
2.	Kernel SHAP	MISC	Accurate	MoRF
3.	LIME-NER	MISC	Accurate	LeRF
4.	Kernel SHAP	MISC	Accurate	LeRF
5.	LIME-NER	MISC	Inaccurate	MoRF
6.	Kernel SHAP	MISC	Inaccurate	MoRF
7.	LIME-NER	MISC	Inaccurate	LeRF
8.	Kernel SHAP	MISC	Inaccurate	LeRF
9.	LIME-NER	LOC	Accurate	MoRF
10.	Kernel SHAP	LOC	Accurate	MoRF
11.	LIME-NER	LOC	Accurate	LeRF
12.	Kernel SHAP	LOC	Accurate	LeRF
13.	LIME-NER	LOC	Inaccurate	MoRF
14.	Kernel SHAP	LOC	Inaccurate	MoRF
15.	LIME-NER	LOC	Inaccurate	LeRF
16.	Kernel SHAP	LOC	Inaccurate	LeRF

4.5.3 Model improvement insights

Post analysis of all the explanations from Kernel SHAP, it is observed that very few of spelling and context related features are ever used in explanation. It is observed that Kernel SHAP used any such feature only in inaccurate 'MISC' entity predictions. As LIME-NER does not provide explanation based on raw features, instead it provides explanations based on words, this insight is not available form LIME-NER explanations.

Taking the above insight into account, the study modifies the NER model in two steps and the performance of modified NER models are evaluated based on the sequence F1 score for both test-a and test-b data sets.

In the first step, the number of spelling and context related features are reduced from NER model input, keeping the model architecture almost same. In the spelling and context feature creation step, as discussed in section 4.2.2, the number of context words set to 0 (i.e. no previous and next word considered). This reduced the dimension of auxiliary input that is directly fed to the CRF layer. The only change in the BI-LSTM-CRF model architecture is highlighted in Figure 25. This also reduced the trainable parameters in the CRF layer as indicated in Table 22.

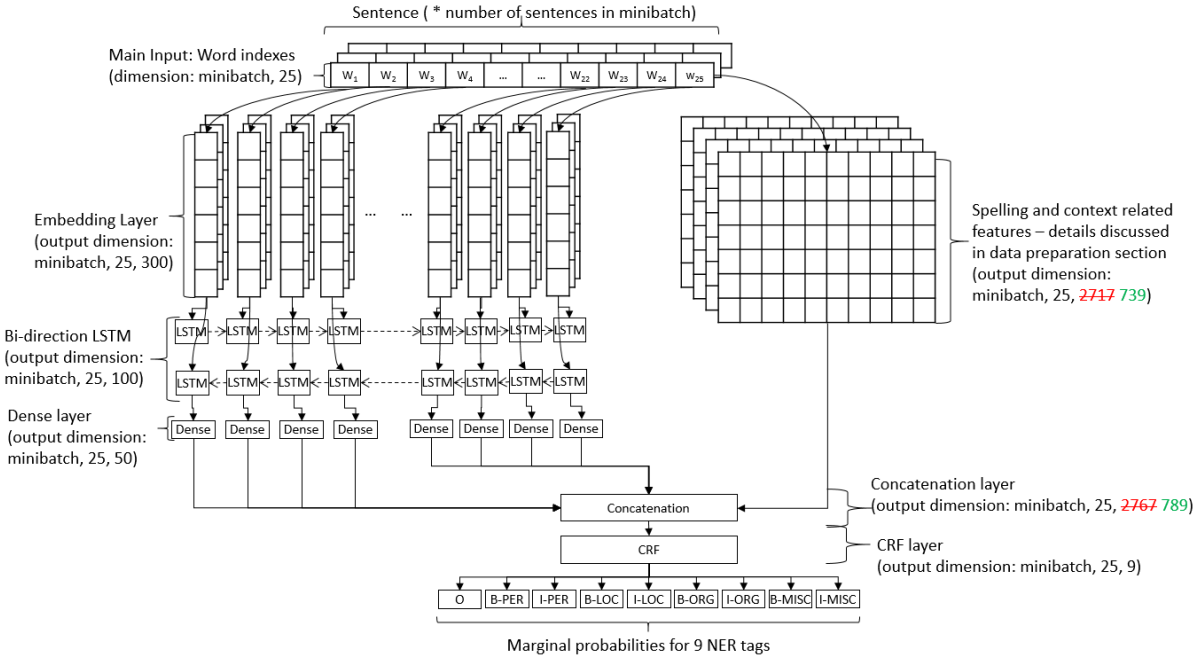

Figure 25: BI-LSTM-CRF architecture for NER model after first step modification

In the second step of modification, spelling and context related features are completely discarded from the BI-LSTM-CRF model. Due to this change, the auxiliary input layer, and the concatenation layer to concatenate dense layer output with spelling and context related feature input are not required anymore. Dense layer output is directly fed to the CRF layer instead. Figure 26 highlights the omissions from initial BI-LSTM-CRF architecture.

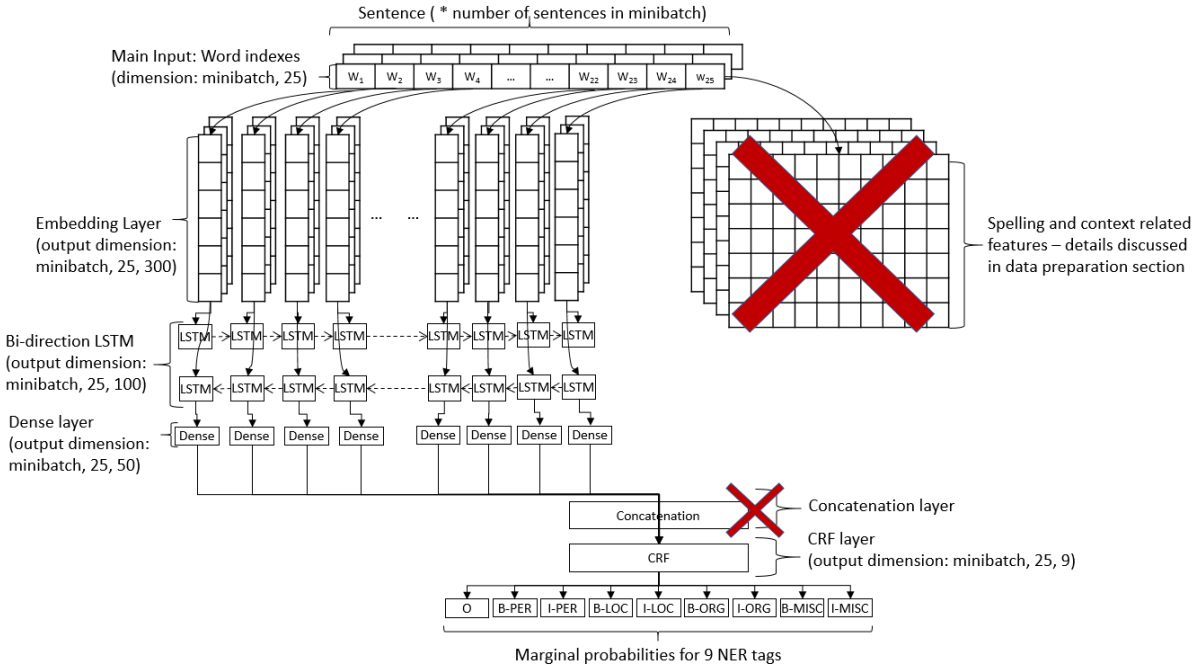

Figure 26: BI-LSTM-CRF architecture for NER model after second step modification

In summary, these modifications in the NER model only impacts the input dimension to the CRF layer and therefore reduces the trainable parameters for that layer. This does not change the output dimension of the CRF layer as the model is still trained on 9 entity tags. The changes in the CRF layer is summarised in Table 22.

Table 22: CRF layer changes in each step of modification

	Initial NER model	NER model after first modification	NER model after second modification
Input dimension to CRF layer	Minibatch size x 25 x 2764	Minibatch size x 25 x 789	Minibatch size x 25 x 50
Trainable parameters in CRF layer	24984	7209	558
Output dimension of CRF layer	Minibatch size x 25 x 9	Minibatch size x 25 x 9	Minibatch size x 25 x 9

Both modified NER models are trained using the same hyperparameters that are used to train the initial NER model as discussed in section 4.3.2.

4.6 Resources

4.6.1 Hardware resources

In this study, all the experiments are performed on a personal computer with the following configurations:

- Intel Core i5-850U CPU, 1.6 GHz, 4 cores
- 8 GB DDR RAM
- Windows 10 64-bit OS
- NVIDIA GeForce MX-150 GPU with 2 GB dedicated memory

4.6.2 Software resources

The experiments in this study are implemented using Python 3.6.9. Apart from python, pre-trained GloVe embedding (with 300 dimensions) is used as discussed in section 0.

Python Keras library is used to build the BI-LSTM-CRF model. Keras is an open-source deep-learning modelling framework that used TensorFlow in the backend.

Python packages - lime and shap are used to generate explanations. Some of the important python packages with versions are mentioned in Table 23.

Table 23: Python packages

Name	Version
keras-gpu	2.2.4
keras-preprocessing	1.1.0
lime	0.2.0.0
numpy	1.16.4
pandas	0.25.1
python	3.6.9
scikit-learn	0.21.2
scipy	1.2.1
seaborn	0.10.1
seqeval	0.0.12
shap	0.34.0
tensorflow-gpu	1.12.0

4.7 Summary

Design and implementation in the study can be broadly divided into four steps. In the first step, the data is processed to make it usable for BI-LSTM-CRF model building. In this step, the feature creation for CRF layer is memory and compute-intensive task.

In the next step, the NER model building using BI-LSTM-CRF is necessary to create the base model for explanation. Though it was required to achieve a reasonable accuracy for the NER model, not a lot of effort is spent in tuning the model as that is not the primary focus of this study.

Explanation generation is one of the primary focus of the study. While earlier literature is available on how to generate explanation for NER models using LIME (Villarroya, 2018), there was no work done to do the same using Kernel SHAP. Therefore, explanation generation using Kernel SHAP is the most important task in this study.

In the final step, the explanations generated from LIME and Kernel SHAP is compared based on multiple evaluation metrics. While AOPC metric was already used in (Villarroya, 2018), other metrics are novel to this study and needed significant experiment and implementation efforts.

CHAPTER 5: RESULTS AND DISCUSSIONS

5.1 Introduction

This section will discuss the results and performance metrics generated form all the experiments outlined in the previous chapter. The section will first discuss the results from the BI-LSTM-CRF model that is built for NER prediction. As the focus of the study is the local explanation of the NER model using LIME and Kernel SHAP, the study will first discuss how to interpret the explanations from these two methods and then will compare these explanation based on qualitative and quantitative evaluation metrics as discussed in section 4.5. Finally, this section will discuss how explanation can be useful finding insights that can help in improving the model performance.

5.2 BI-LSTM-CRF model performance metrics

The baseline NER model built using BI-LSTM-CRF architecture is evaluated using Sequence F1 score. Table 24 shows the sequence F1 score for the NER model prediction on test-a data set.

Table 24: NER model sequence F1 score on the test-a dataset

Entity	Sequence F1 score
LOC	0.89
PER	0.78
ORG	0.78
MISC	0.76
Macro average	0.82

The macro average sequence F1 score suggests that the model is reasonably accurate on test-a data set. As the focus of the study is to perform local explanation for best performing and worst performing entity, ‘LOC’ and ‘MISC’ entities are selected for the generation and evaluation of explanations.

5.3 Interpretation of explanations

LIME and Kernel SHAP presents the explanations in different ways. While LIME implementation has visualization for text by design, SHAP implementation does not have that

feature. In this section, the study will discuss how to interpret the explanations generated from both the methods and compare them in subsequent sections.

5.3.1 LIME-NER explanation

For LIME-NER explanation, the study has experimented with two methods. In one of the methods, experiments are set up exactly as described by (Villarroya, 2018). That study has discussed (see section 3.3.1 for details) two improvements on original LIME implementation, one related to using entity level prediction probability instead of using B and I tag probability, another related to perturbing entity words together instead of perturbing them individually. Table 25 provides details on the sentence, words, and entity for explanations that are presented next.

Table 25: Instance details for explanation

Sentence	The Republic of Ireland beat Liechtenstein 5-0 (halftime 4-0) in a World Cup soccer European group 8 qualifier on Saturday .
Entity words for explanation	World Cup
Predicted entity	MISC
Ground truth entity	MISC
Entity level prediction probability	1

Figure 27 shows the explanation for the sentence and word mentioned in Table 25 using LIME-NER with both the improvements discussed in (Villarroya, 2018). The figure suggests that entity level prediction probability for the words 'World Cup' is 1 and this is a correct prediction as well according to Table 25 (predicted entity and ground truth entity is same). The feature importance plot for MISC entity highlights that the word 'World' has the highest contribution for this prediction. The text with highlighted words towards the bottom of the figure suggests the same and able to express the position of the important words in the sentence which helps one to comprehend the explanation easily. However, it is observed that the MISC entity has two words; the second word, 'Cup' has not been assigned any importance.

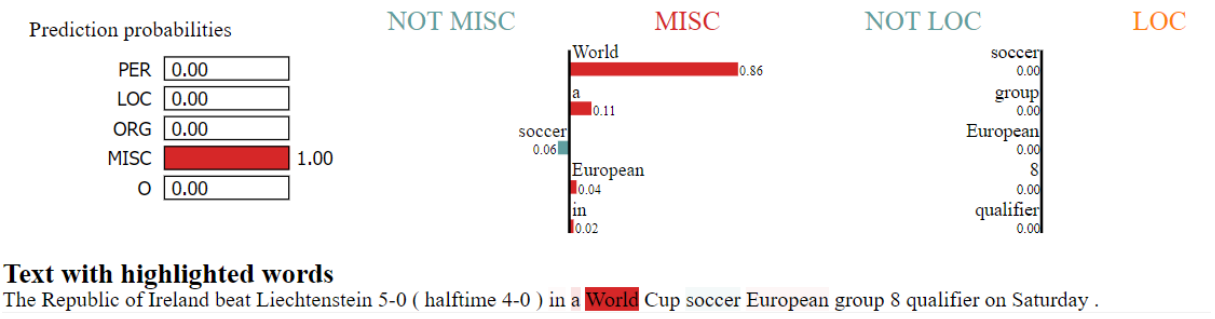

Figure 27: LIME-NER explanation with both properties

Figure 28 shows explanation for the sentence and word mentioned in Table 25 using the other variation of LIME-NER. This variation of LIME-NER uses the entity level prediction probabilities but does not restrict perturbation of entity words together, i.e. it allows perturbation of entity words individually. Here, it is observed that the word 'Cup' has the highest importance and the word 'World' also have positive importance in the prediction. This seems a more reasonable explanation as both words are part of the entity.

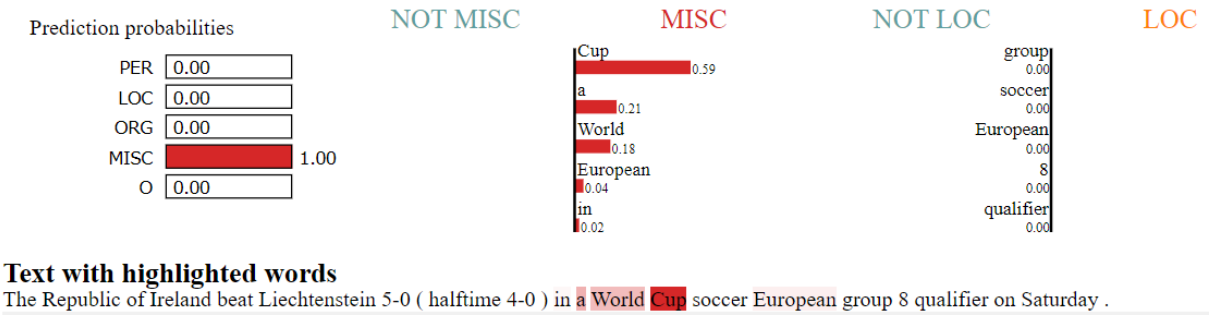

Figure 28: LIME-NER with only one property

5.3.2 Kernel SHAP explanation

For the same sentence, word and entity in Table 25, Kernel SHAP explanations are shown from Figure 29 to Figure 32. The force plot in Figure 29 suggests that the words 'Cup', 'a', 'World, 'European' are the contributing words in the descending order of importance. The summary plot for predicted entity 'MISC' at Figure 30 conveys the same explanation in a different way. The summary plot also mentions the word 'European' clearly that is not visible in force plot. In this case, the force plot does not have the blue colour arrows that indicate the features that decreased the probability as shown in Figure 21. The reason for that is the prediction probability is 1, so there are no forces present to reduce the probability. That is the same reason the summary plot does not have any feature with negative SHAP values. All the features visible in the force plot and summary plot are enriched as discussed in section 4.4.3. Therefore, the format of the visible features is 'word>word position in sentence>POS tag>predicted entity tag'. The

study observes the summary plot does not provide any additional information compared to force plot except clearly mentioning the fourth contributing word feature. Therefore, from this point onwards the study has used a summary plot if any of the feature names are not completely visible in force plot.

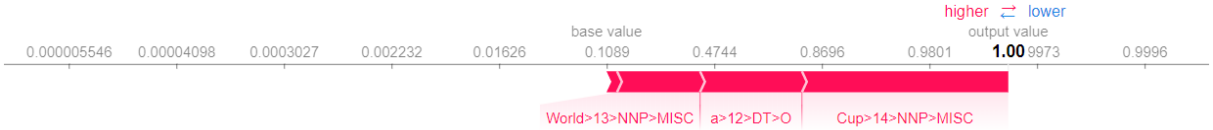

Figure 29: Kernel SHAP force plot

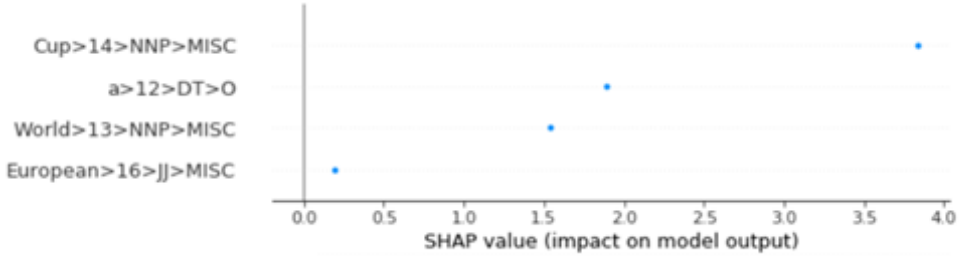

Figure 30: Kernel SHAP summary plot for the predicted entity

The subsequent visualisations offered by Kernel SHAP adds value to the explanation as that is not directly available in LIME. The multioutput decision plot in Figure 31 shows how different features impacted the prediction probability of all entity tags. This figure shows that the word features below the word ‘European’ does not have an impact on the prediction of ‘MISC’ as the entity (features highlighted with a red box in Figure 31). The word ‘European’ show first positive contribution towards entity tag prediction as ‘MISC’ (feature highlighted with an amber box in Figure 31). However, until this point, probability of entity ‘O’ is higher. The word ‘scorer’ does not have any direct impact either on ‘MISC’ entity tag prediction or decreasing probability for ‘O’ entity tag. However, it has a negative contribution towards other entity tags (PER, ORG and LOC) being predicted for the words. This signifies that the word ‘scorer’ has some contribution to overall prediction. Finally, the words ‘a’, ‘World’ and ‘Cup’ has pushed the probability for ‘MISC’ tag prediction over the probability of ‘O’ tag prediction (feature highlighted with a green box in Figure 31).

The NER prediction probabilities suggest that the next probable entity tag after ‘MISC’ is ‘LOC’ (probability 0.0001). However, the multioutput decision plot suggests that the words

part of the entity is the main reason for final prediction of ‘MISC’ and not ‘O’. The entity tag ‘LOC’ was never a contender for the prediction. Understanding of the fact that entity ‘O’ is more probable NER prediction if only the surrounding words of the entity words are considered (i.e. in absence of words ‘a’, ‘World’ and ‘Cup’) is unique to Kernel SHAP explanation.

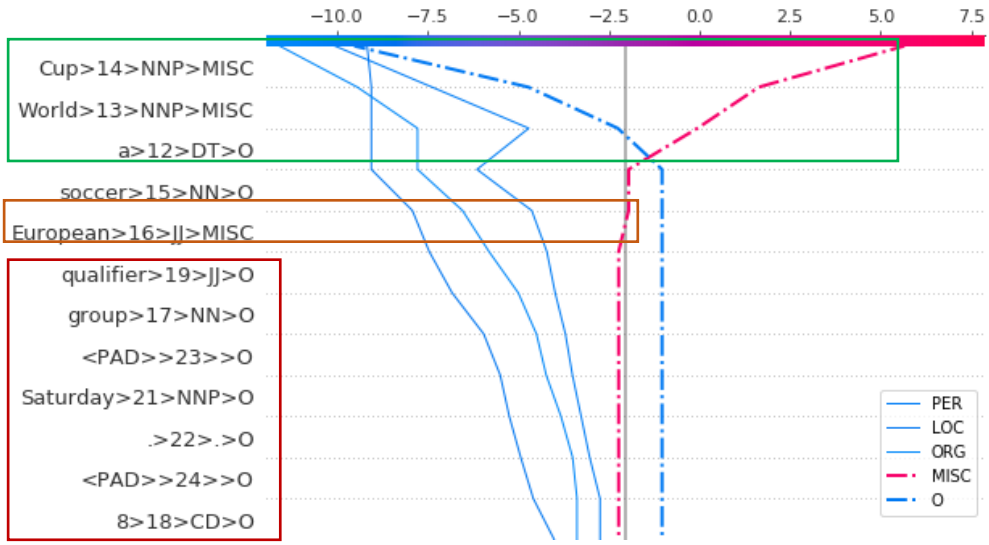

Figure 31: Kernel SHAP multioutput decision plot

The summary plot in Figure 32 shows the influence of each word for predicting any of the entity tags for the sentence and word combination in Table 25. This presents the absolute (i.e. unsigned) influence of the word. It is observed that words featuring in the plot are the same in the multioutput decision plot in Figure 31. The multioutput decision plot is showing the direction of influence (positive or negative) and the summary plot is showing the total influence of a word on the prediction. As the study is focused on explaining prediction for individual entities, this summary plot is not used to observe other Kernel SHAP explanations unless it is required to highlight any important point.

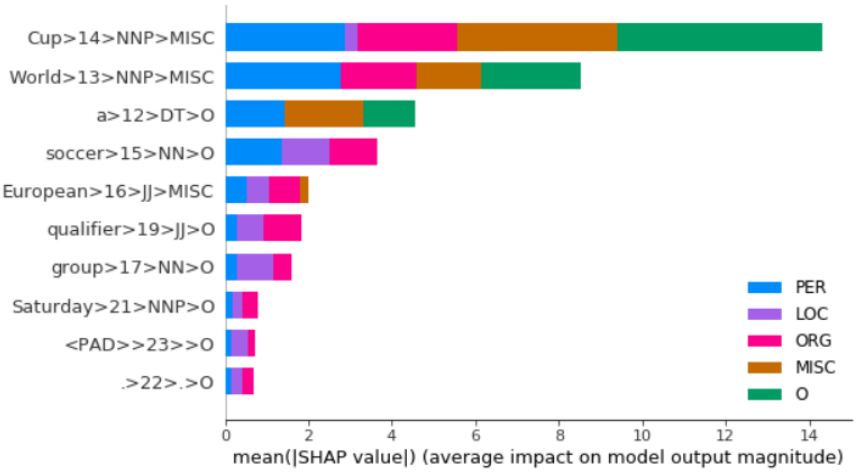

Figure 32: Kernel SHAP summary plot for all entities

Comparison of generated explanation from Kernel SHAP with LIME-NER as proposed by (Villarroya, 2018) and LIME-NER only with entity level predictions indicates that the explanation outcome for first LIME-NER method is subpar. It missed highlighting contributing words that are common in explanations generated by second LIME-NER and Kernel SHAP explanation methods. Therefore, the study uses the first improvement proposed in (Villarroya, 2018), i.e. use the entity level prediction probabilities so that the explanations are generated for an entity as a whole instead of B and I tags, but does not consider using the updated LIME algorithm for perturbing entity words together. For all future reference to LIME-NER in this study points to the above-mentioned method.

5.4 Visual comparison of explanations

This section will present the explanations generated from LIME-NER and Kernel SHAP for different scenarios mentioned in its subsections. The focus of the section will be to visually compare them and form an intuition whether any method is better than the other in different scenarios.

This section compares the accurate and inaccurate LOC and MISC predictions for known and unknown entity words. Known entity word indicates that the word which is part of the entity is available in training vocabulary.

5.4.1 Accurate LOC prediction with known entity words

Explanations are generated for the sentence, word and entity provided in Table 26 using LIME-NER and Kernel SHAP.

Table 26: Instance details for accurate LOC prediction with known entity words

Sentence	Scores : Australia 228-9 in 50 overs , Sri Lanka 232-6 in 45.5 overs .
Entity words for explanation	Sri Lanka
Predicted entity	LOC
Ground truth entity	LOC
Entity level prediction probability	0.98

Both LIME-NER explanation in Figure 33 and Kernel SHAP force plot in Figure 34 shows the word ‘Lanka’ has the highest positive contribution followed by the word ‘Sri’. Few words are having very little negative influence on the ‘LOC’ entity prediction, e.g. ‘overs’, ‘Scores’.

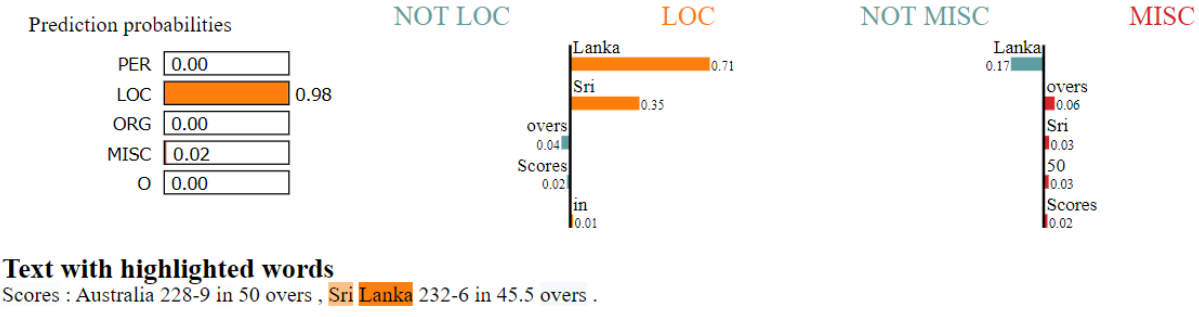

Figure 33: LIME-NER explanation for accurate LOC prediction with known entity words

Figure 34: Kernel SHAP force plot for accurate LOC prediction with known entity words

The multioutput decision plot shows that even if ‘MISC’ has second-highest probability after ‘LOC’, the prediction would have been ‘O’ if the words ‘Sri’ and ‘Lanka’ is replaced with words that are unknown in training. This can answer the question of why

‘LOC’ and not ‘MISC’ or ‘O’ is predicted for the words which is one of the objectives of this study.

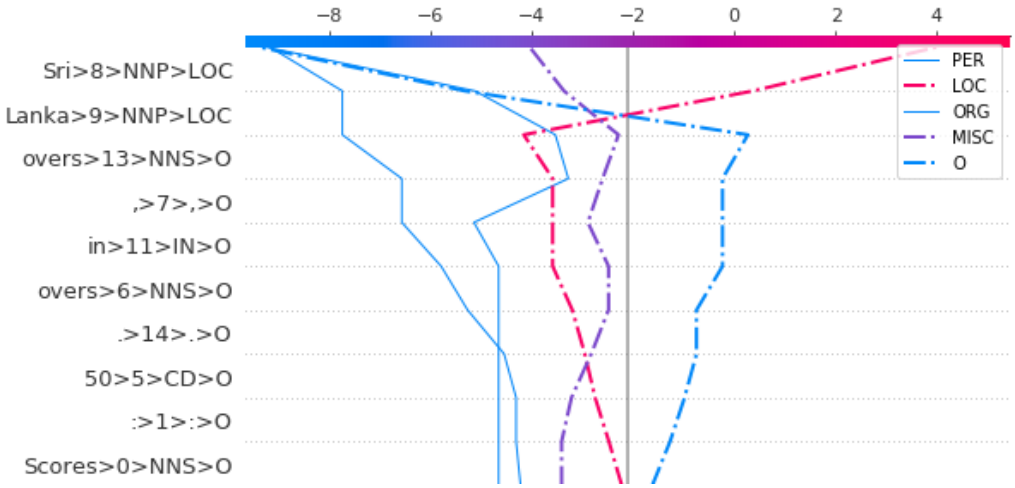

Figure 35: Kernel SHAP multioutput decision plot for accurate LOC prediction with known entity words

5.4.2 Accurate LOC prediction with unknown entity words

Explanations are generated for the sentence, word and entity provided in Figure 27 using LIME-NER and Kernel SHAP. In this case, the entity word is not part of the vocabulary that is used to train the NER model.

Table 27: Instance details for accurate LOC prediction with unknown entity words

Sentence	TOFTIR , Faroe Islands 1996-08-31
Entity words for explanation	TOFTIR
Predicted entity	LOC
Ground truth entity	LOC
Entity level prediction probability	0.95

LIME-NER explanation presented in Figure 36. This shows that the words ‘Faroe’ and ‘Islands’ appearing after the entity word has the most positive influence on the NER prediction.

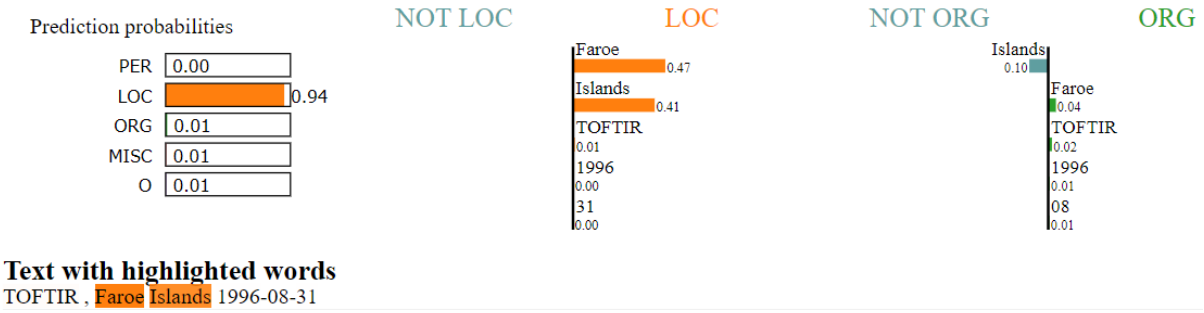

Figure 36: LIME-NER explanation for accurate LOC prediction with unknown entity words

Kernel SHAP force plot in Figure 37 also shows that the feature 'TOFTIR|<UNK>:+3:Islands__word.len()' has a negative impact on the prediction. Though the impact is minor, the feature has a keyword, <UNK>, that needs discussion. As part of the feature enrichment discussed in section 4.4.3, this keyword is added to the word if the word is not part of training vocabulary. The summary plot is presented in Figure 38 to show this feature name clearly. The summary plot also shows the positive impact of the word <PAD> that indicates the dummy word added immediately after the sentence ends.

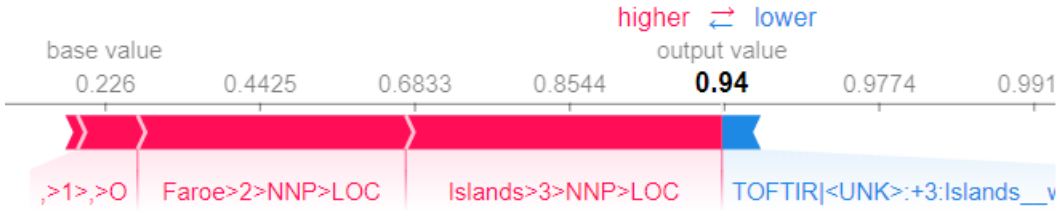

Figure 37: Kernel SHAP force plot for accurate LOC prediction with unknown entity words

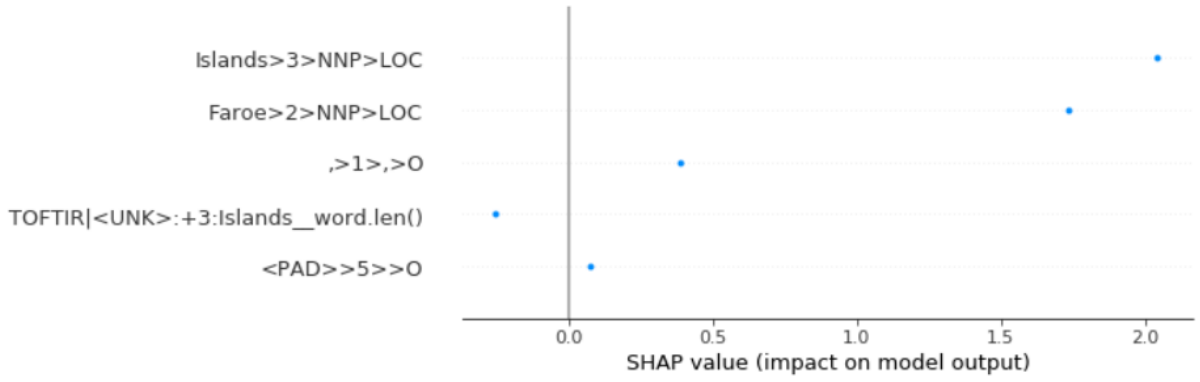

Figure 38: Kernel SHAP summary plot for accurate LOC prediction with unknown entity words

The multioutput decision plot at Figure 39 indicates that absence of the words 'Faroe' and 'Islands' (which are part of the training vocabulary) could have forced the entity prediction to be 'O' even if as per the prediction probability of NER model the second-highest probability is for 'ORG'.

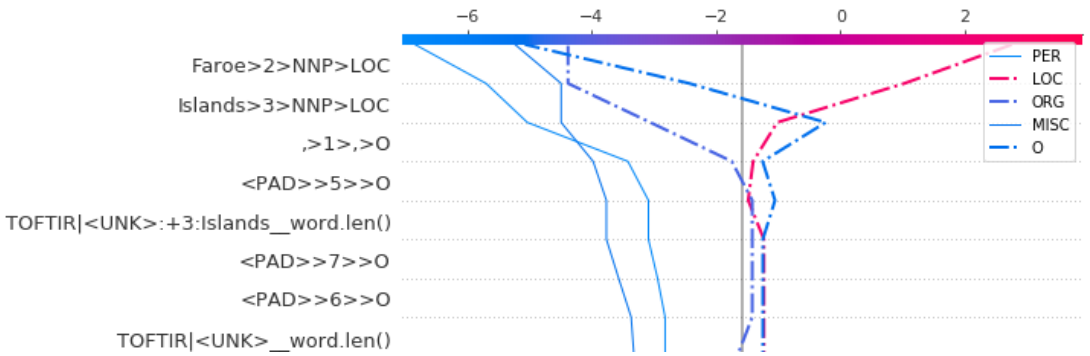

Figure 39: Kernel SHAP multioutput decision plot for accurate LOC prediction with unknown entity words

5.4.3 Inaccurate LOC prediction with known entity words

Explanations are generated for the sentence, word and entity provided in Table 28 using LIME-NER and Kernel SHAP. In this case prediction of the words as 'LOC' is not accurate with respect to the ground truth.

Table 28: Instance details for inaccurate LOC prediction with known entity words

Sentence	I think he got a little tight at a couple of moments , said Washington .
Entity words for explanation	Washington
Predicted entity	LOC
Ground truth entity	PER
Entity level prediction probability	0.98

LIME-NER explanation in Figure 40 and Kernel SHAP force plot in Figure 41 shows that the entity word 'Washington' has the highest influence in predicting the NER tag for the word as 'LOC'. Understandably this influence is because the word 'Washington' is present in training vocabulary as well as in pre-trained GloVe embedding as location. They also show that the word 'said' that appeared immediately before the entity word has a minor positive influence on the word being predicted as location.

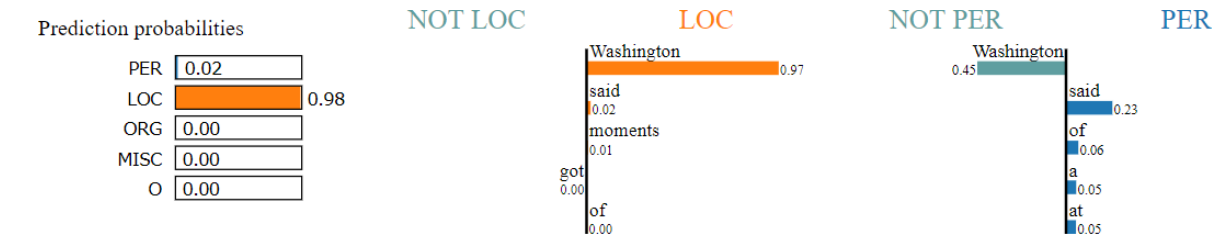

Figure 40: LIME-NER explanation for inaccurate LOC prediction with known entity words

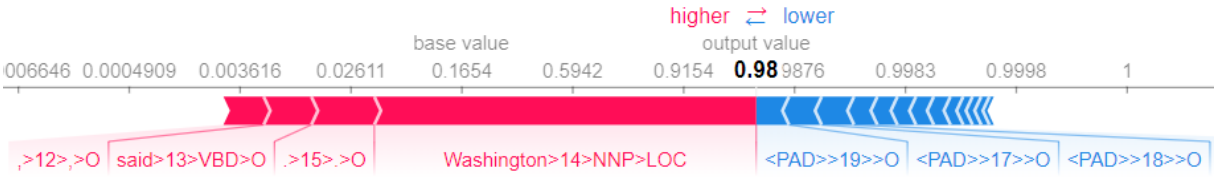

Figure 41: Kernel SHAP force plot for inaccurate LOC prediction with known entity words

LIME-NER explanation in Figure 40 also shows that the word ‘said’ has a positive influence in raising the prediction probability of ‘PER’ that is the ground truth entity tag for the word. However, it does not answer whether the presence of the word ‘said’ is sufficient for the entity word to be predicted as ‘PER’ if the word ‘Washington’ is replaced with an unknown (to training) word. The multioutput decision plot in Figure 42 shows that the propensity of the predicting ‘PER’ tag for the entity word increased over ‘LOC’ with the word ‘said’ if the entity word ‘Washington’ is not considered.

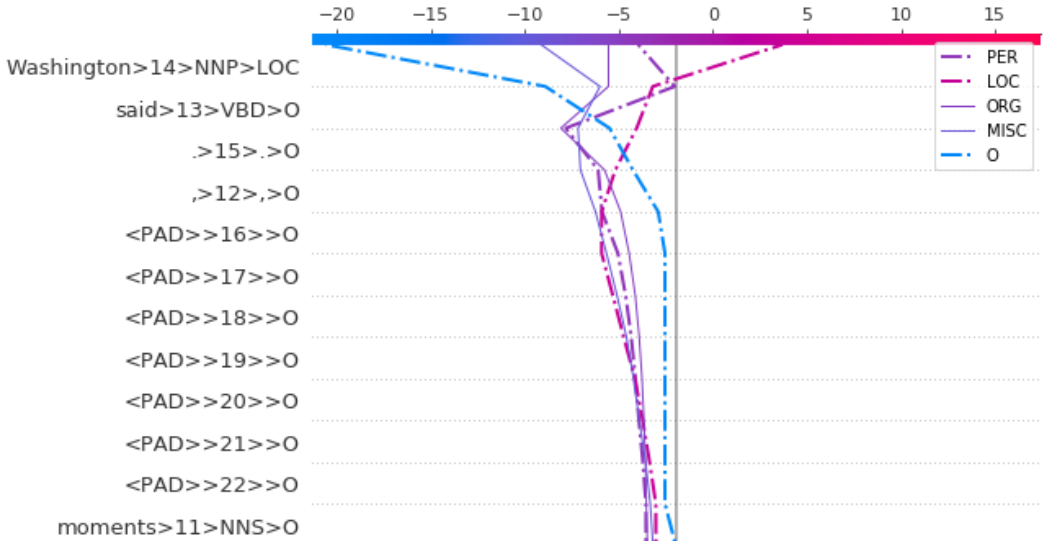

Figure 42: Kernel SHAP multioutput decision plot for inaccurate LOC prediction with known entity words

5.4.4 Inaccurate LOC prediction with unknown entity words

Explanations are generated for the sentence, word and entity provided in Table 29 using LIME-NER and Kernel SHAP. In this case prediction of the words as 'LOC' is not accurate with respect to the ground truth and the entity word is not part of training vocabulary.

Table 29: Instance details for inaccurate LOC prediction with unknown entity words

Sentence	Brokers said blue chips like IDLC , Bangladesh Lamps , Chittagong Cement and Atlas Bangladesh were expected to rise .
Entity words for explanation	IDLC
Predicted entity	LOC
Ground truth entity	ORG
Entity level prediction probability	0.68

The LIME-NER explanation in Figure 43 and Kernel SHAP force plot in Figure 44 shows that the word 'Bangladesh' has the highest positive influence for the prediction of 'LOC' tag. This is the following word of the entity word 'IDLC' and the word 'Bangladesh' is part of the training vocabulary. Both the plots show the word 'like' also have a positive influence. The force plot shows two more words, 'Cement' and 'Atlas', having a positive influence on the prediction of 'LOC'. Notably, the second-highest NER prediction probability is for 'PER' entity and not 'ORG' entity which is the ground truth entity tag.

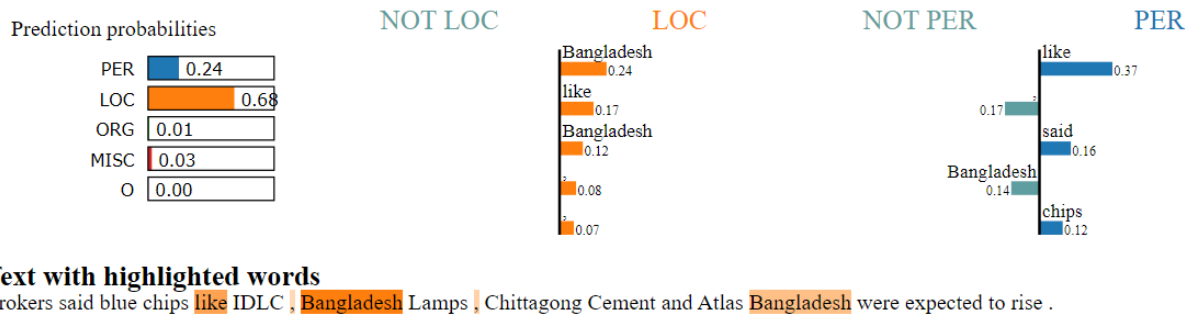

Figure 43: LIME-NER explanation for inaccurate LOC prediction with unknown entity words

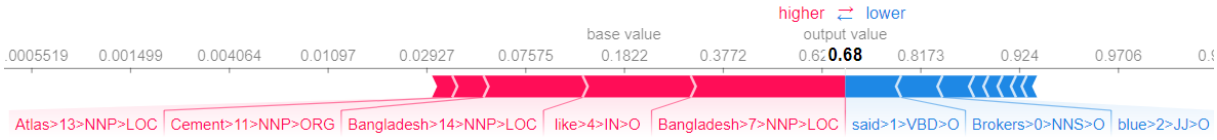

Figure 44: Kernel SHAP force plot for inaccurate LOC prediction with unknown entity words

The multioutput decision plot in Figure 45 clearly shows that only the word 'like' has a positive influence for 'ORG' tag, but it has a similar amount of influence for 'LOC' tag as well.

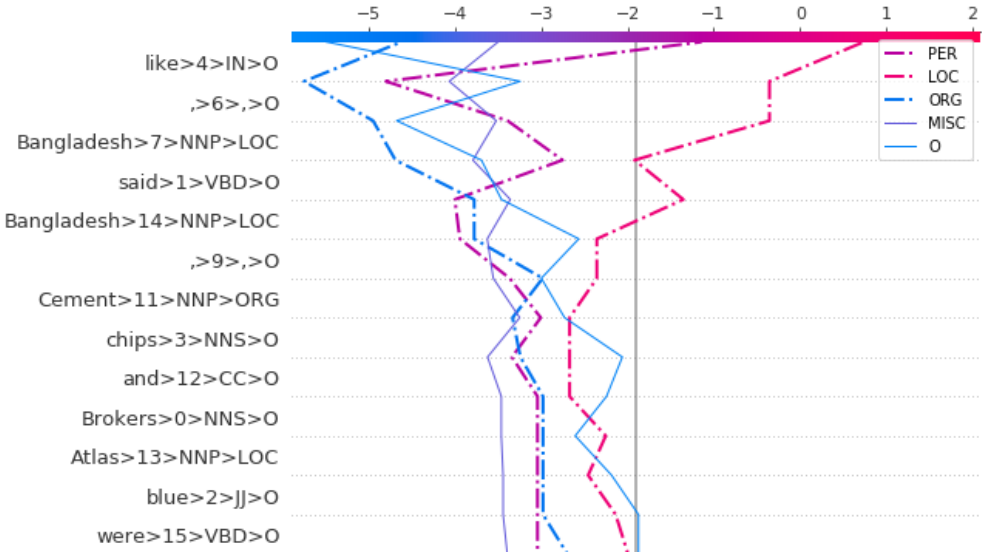

Figure 45: Kernel SHAP multioutput decision plot for inaccurate LOC prediction with unknown entity words

5.4.5 Accurate MISC prediction with known entity words

An example explanation satisfying this scenario is already discussed in section 5.3. There were no other instances where MISC entity with multiple words predicted with more than 0.95 probability score for test-a data set.

5.4.6 Accurate MISC prediction with unknown entity words

Explanations are generated for the sentence, word and entity provided in Table 30 using LIME-NER and Kernel SHAP. In this case entity word is not part of training vocabulary.

Table 30: Instance details for accurate MISC prediction with unknown entity words

Sentence	RUGBY UNION - ENGLISH , SCOTTISH AND WELSH RESULTS .
Entity words for explanation	WELSH
Predicted entity	MISC
Ground truth entity	MISC
Entity level prediction probability	0.47

The LIME-NER explanation in Figure 46 and Kernel SHAP force plot in Figure 47 shows the same word for positive influence but in a different order. Only the most influential word, 'English', is same in both the explanations.

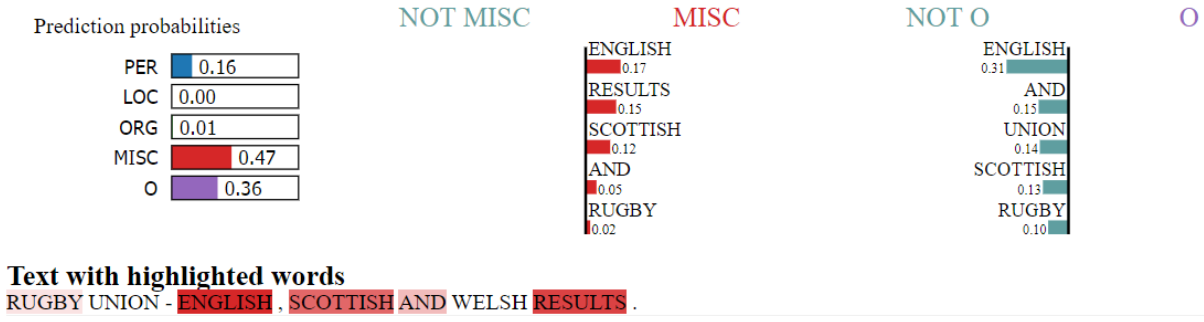

Figure 46: LIME-NER explanation for accurate MISC prediction with unknown entity words

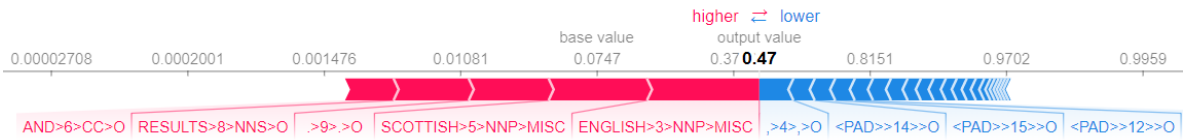

Figure 47: Kernel SHAP force plot for accurate MISC prediction with unknown entity words

The multioutput decision plot in Figure 48 indicates that the NER model had difficulty in differentiating between the prediction of 'MISC', 'O' and 'PER' entity for the word. This is also visible in probability scores presented by the LIME-NER explanation. However multioutput decision plot clearly points out that the word 'RESULTS' which comes after the entity word 'WELSH' has changed the prediction from 'PER' to 'MISC' tag.

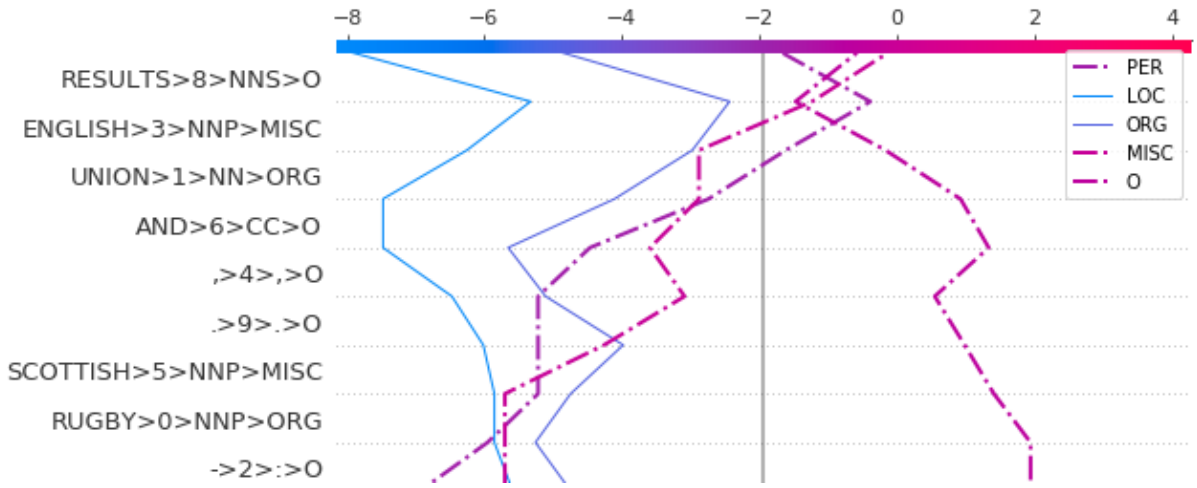

Figure 48: Kernel SHAP multioutput decision plot for accurate MISC prediction with unknown entity words

5.4.7 Inaccurate MISC prediction with known entity words

Explanations are generated for the sentence, word and entity provided in Table 31 using LIME-NER and Kernel SHAP. In this case prediction of 'MISC' entity for the entity word is inaccurate while entity word is part of training vocabulary.

Table 31: Instance details for inaccurate MISC prediction with known entity words

Sentence	elected to bat against Sri Lanka in the third day-night limited
Entity words for explanation	day-night
Predicted entity	MISC
Ground truth entity	O
Entity level prediction probability	0.98

The LIME-NER explanation in Figure 49 and Kernel SHAP force plot in Figure 50 shows that the entity word 'day-night' followed by 'third' have the highest positive influence in 'MISC' entity prediction. The other words highlighted by both the explanations are the same but in a different order.

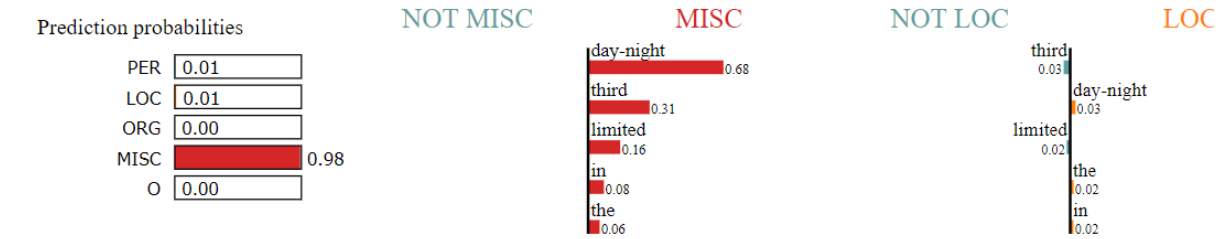

Figure 49: LIME-NER explanation for inaccurate MISC prediction with known entity words

Figure 50: Kernel SHAP force plot for inaccurate MISC prediction with known entity words

The multioutput decision plot in Figure 51 indicates the words ‘third’ followed by ‘day-night’ have influenced the prediction of ‘O’ tag negatively as well.

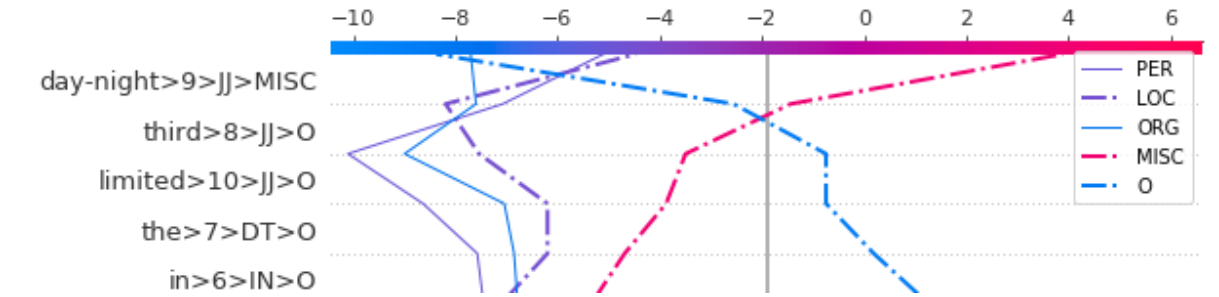

Figure 51: Kernel SHAP multioutput decision plot for inaccurate MISC prediction with known entity words

5.4.8 Inaccurate MISC prediction with unknown entity words

Explanations are generated for the sentence, word and entity provided in Table 32 using LIME-NER and Kernel SHAP. In this case prediction of ‘MISC’ entity for the entity word is inaccurate while entity word is not part of training vocabulary.

Table 32: Instance details for inaccurate MISC prediction with unknown entity words

Sentence	Missing German girl found alive at Dutch campsite .
Entity words for explanation	Missing
Predicted entity	MISC
Ground truth entity	O
Entity level prediction probability	0.93

The LIME-NER explanation in Figure 52 and Kernel SHAP force plot in Figure 53 shows the same words influencing 'MISC' prediction. The order of influence is also the same for both explanations.

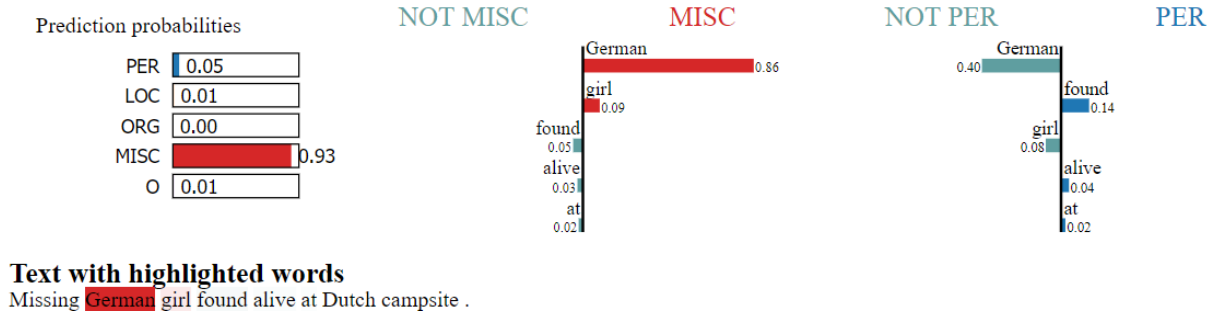

Figure 52: LIME-NER explanation for inaccurate MISC prediction with unknown entity words

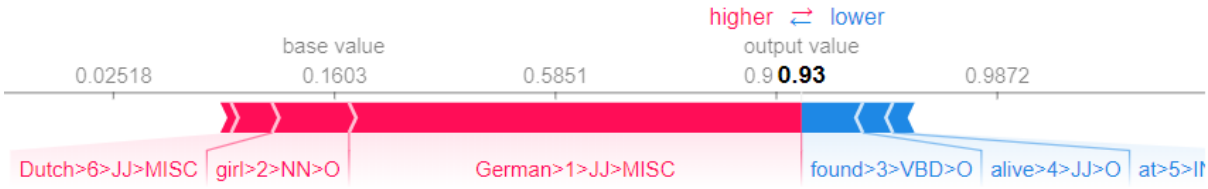

Figure 53: Kernel SHAP force plot for inaccurate MISC prediction with unknown entity words

The multioutput decision plot in Figure 54 indicates the word ‘German’ that is having the most positive influence on ‘MISC’ prediction is also predicted as ‘MISC’. Absence of that word would have predicted ‘PER’ entity for the word ‘Missing’

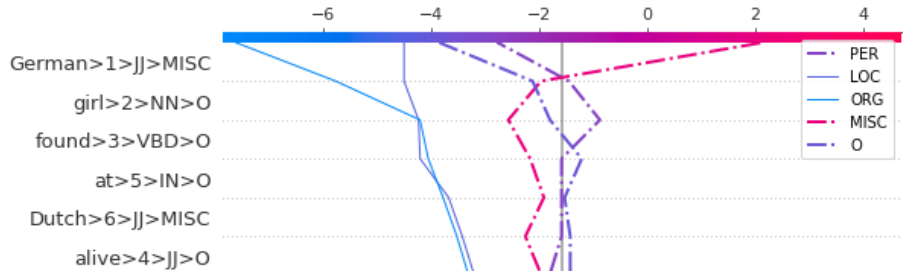

Figure 54: Kernel SHAP multioutput decision plot for inaccurate MISC prediction with unknown entity words

After analysing the explanation from both LIME-NER and Kernel SHAP in this section it is observed that if entity words are unknown to the vocabulary, then those words do not seem to have any influence in the NER prediction. In general, it is also observed that the spelling and context related features are not appearing to be top influencers for the NER predictions.

5.5 Explanation evaluation metrics

The explanations generated from LIME-NER and Kernel SHAP is evaluated based on the evaluation metrics as discussed in section 4.5.

5.5.1 Qualitative evaluation

This subsection will discuss the quality of explanations as derived using the methods from section 4.5.1.

- Accuracy – The accuracy of explanations is understood by observing explanation for the instances where NER model prediction is accurate, but the prediction probability is very close to another entity class.

  Table 33 contains the details of the instance to analyse the accuracy of explanation where ‘LOC’ entity is predicted over ‘PER’. In this case, entity words are part of training vocabulary.

Table 33: Instance details for LOC prediction with known entity words

Sentence	N. Korea urges S. Korea to return war veteran .
Entity words for explanation	N. Korea
Predicted and Ground truth entity	LOC
Next possible entity as per prediction probability	PER
Entity level prediction probability for LOC	0.48

The LIME-NER explanation in Figure 55 and Kernel SHAP force plot in Figure 56 clearly shows that the word 'Korea' has a most positive influence and 'N.' has a most negative influence in 'LOC' prediction. The LIME-NER explanation also shows that 'Korea' has the most negative influence and 'N.' has the most positive influence in predicting 'PER'.

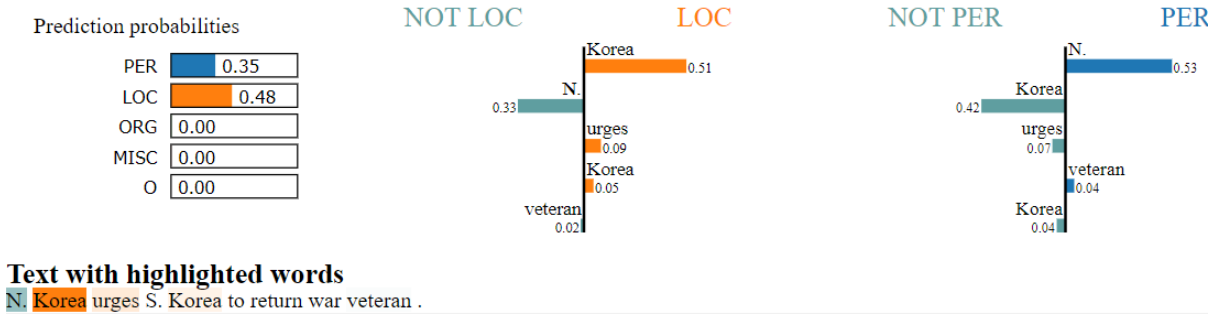

Figure 55:LIME-NER explanation for LOC prediction with known entity words

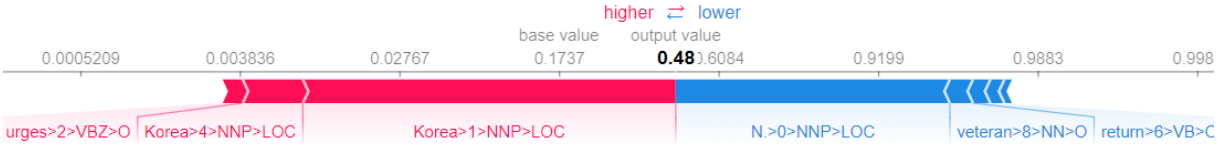

Figure 56: Kernel SHAP force plot for LOC prediction with known entity words

The multioutput decision plot from Kernel SHAP in Figure 57 highlights that 'Korea' has the most negative influence and 'N.' has the most positive influence in predicting 'PER'.

In this scenario, though both LIME-NER and Kernel SHAP explanations seem accurate, the explanation provided by LIME-NER highlighting text in different colours seem more convenient for comprehension.

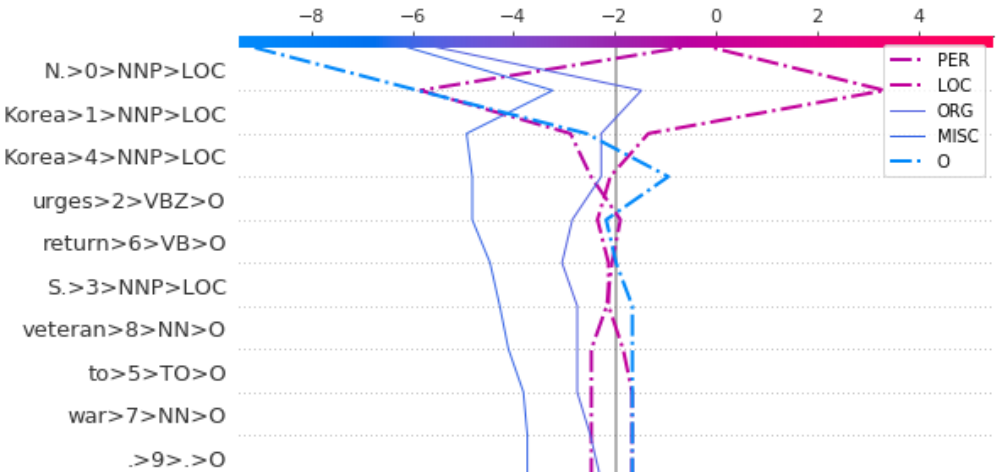

Figure 57: Kernel SHAP multioutput decision plot for LOC prediction with known entity words

Table 34 contains the details of the instance to analyse the accuracy of explanation where 'LOC' entity is predicted over 'PER'. In this case entity word is not part of training vocabulary.

Table 34: Instance details for LOC prediction with unknown entity words

Sentence	His aides said Arafat would hold the weekly meeting of the Palestinian self-rule Authority 's cabinet in Nablus on Saturday .
Entity words for explanation	Nablus
Predicted and Ground truth entity	LOC
Next possible entity as per prediction probability	PER
Entity level prediction probability for LOC	0.36

LIME-NER explanation in Figure 58 shows that the words 'in' and 'cabinet' that are two preceding words of the entity words have the most positive influence in 'LOC'

prediction. However, these two words also have the most positive influence in ‘PER’ prediction. LIME-NER explanation also shows that the word ‘on’ that comes immediately after the entity word has a negative influence in ‘PER’ prediction but a positive influence in ‘LOC’ prediction.

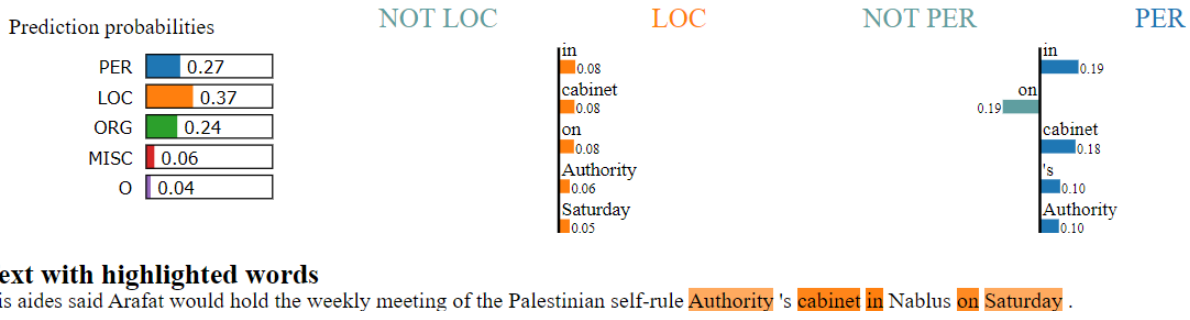

Figure 58: LIME-NER explanation for LOC prediction with unknown entity words

The Kernel SHAP force plot in Figure 59 shows the positive influence in ‘LOC’ prediction similar to LIME-NER explanation. However, this plot does not shed any light on why ‘PER’ is not predicted.

Kernel SHAP multioutput decision plot in Figure 60 shows that the word ‘on’ is the reason for the prediction probability difference between ‘LOC’ and ‘PER’

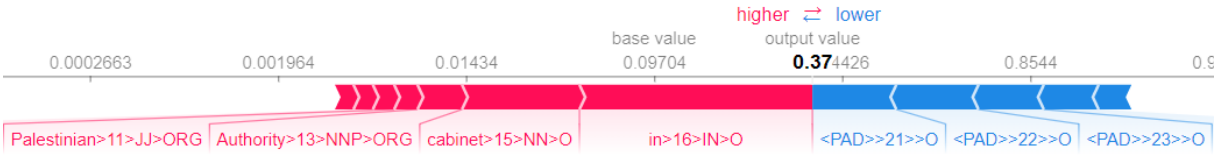

Figure 59: Kernel SHAP force plot for LOC prediction with unknown entity words

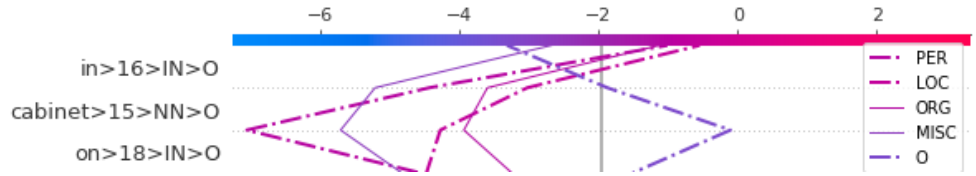

Figure 60: Kernel SHAP multioutput decision plot for LOC prediction with unknown entity words

Table 35 contains the details of the instance to analyse the accuracy of explanation where ‘MISC’ entity is predicted over ‘ORG’. In this case, entity words are part of training vocabulary.

Table 35: Instance details for MISC prediction with known entity words

Sentence	The Canadian market typically closes early on holiday weekends and Canadian financial markets will be closed on Monday for Labour Day .
Entity words for explanation	Labour Day
Predicted and Ground truth entity	MISC
Next possible entity as per prediction probability	ORG
Entity level prediction probability for MISC	0.50

LIME-NER explanation in Figure 61 indicates that the word 'Day' that is the second word in the entity words has a positive influence in predicting 'MISC' entity but negative influence in predicting 'ORG' entity. The first word in entity words, 'Labour', has a more positive influence in predicting 'ORG' as NER tag and less positive influence in predicting 'MISC' as NER tag.

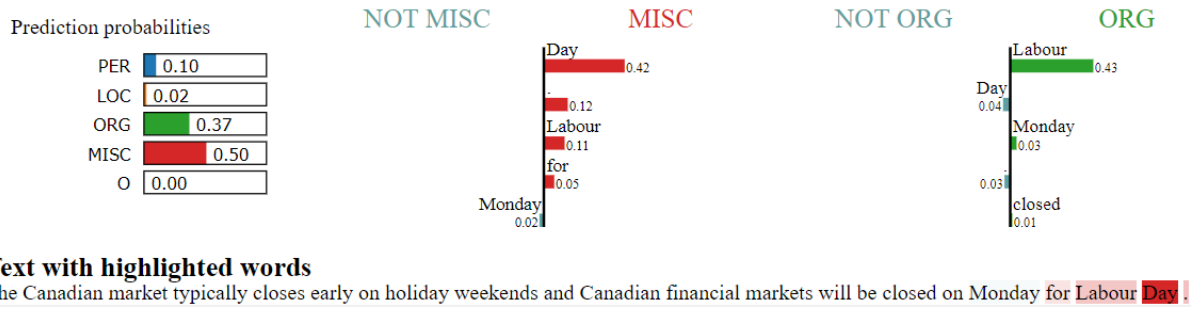

Figure 61: LIME-NER explanation for MISC prediction with known entity words

Kernel SHAP force plot in Figure 62 does not provide any information in terms of why 'MISC' entity is predicted over 'ORG' entity. The Kernel SHAP multioutput decision plot in Figure 63 answers this question like LIME-NER explanation.

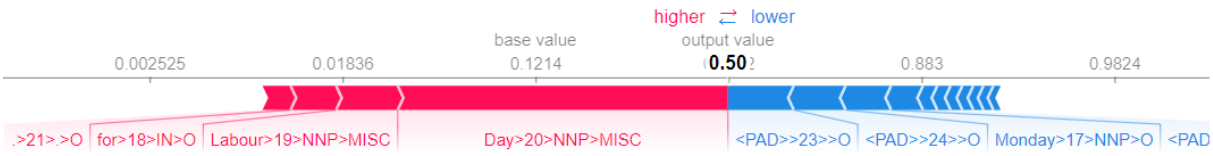

Figure 62: Kernel SHAP force plot for MISC prediction with known entity words

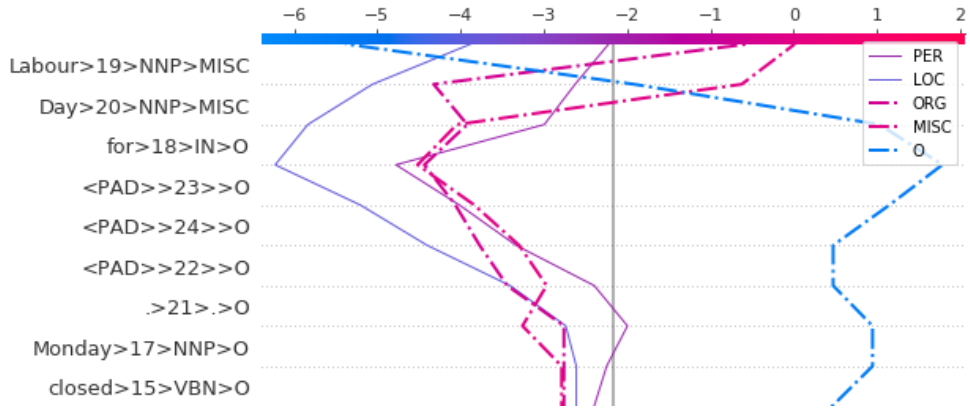

Figure 63: Kernel SHAP multioutput decision plot for MISC prediction with known entity words

Table 36 contains the details of the instance to analyse the accuracy of explanation where 'MISC' entity is predicted over 'PER'. In this case entity word is not part of training vocabulary.

Table 36: Instance details for MISC prediction with unknown entity words

Sentence	GOLF - LEADING SCORES AT GREATER MILWAUKEE OPEN .
Entity words for explanation	GREATER
Predicted and Ground truth entity	MISC
Next possible entity as per prediction probability	PER
Entity level prediction probability for MISC	0.49

LIME-NER explanation in Figure 64 shows that the word 'OPEN' has a positive influence in predicting 'MISC entity and negative influence in predicting 'PER'

entity. It shows that the word ‘AT’ has a positive influence on both ‘MISC’ and ‘PER’, but in the case of ‘MISC’, the influence is more.

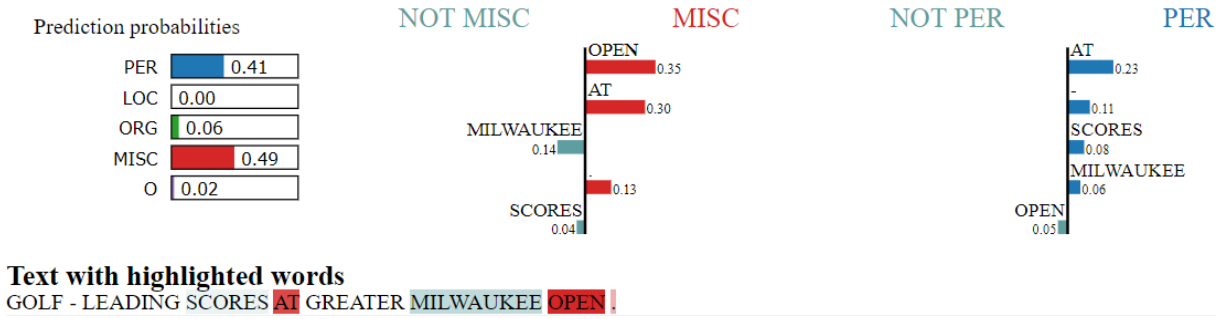

Figure 64: LIME-NER explanation for MISC prediction with unknown entity words

Kernel SHAP fore plot in Figure 65 shows features for positive and negative influence on ‘MISC’ prediction, but is not designed to answer why ‘MISC’ entity is predicted over ‘PER’ entity

Figure 65: Kernel SHAP force plot for MISC prediction with unknown entity words

Kernel SHAP multioutput decision plot in Figure 66 shows that how the word ‘OEPN’ has impacted ‘MISC’ and ‘PER’ entity prediction. Additionally, it shows how the words ‘OPEN’ and ‘AT’ have impacted the prediction of ‘O’ entity (and any other entities).

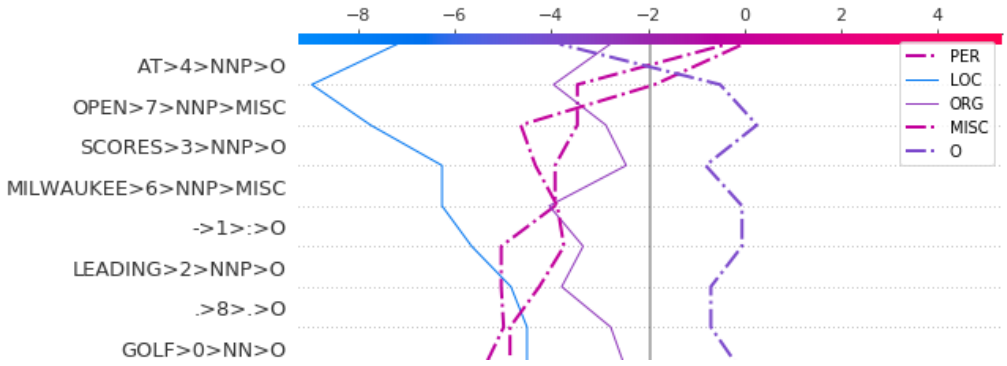

Figure 66: Kernel SHAP multioutput decision plot for MISC prediction with unknown entity words

Analysis of these sample instances from four different scenarios suggests that LIME-NER explanation and Kernel SHAP multioutput decision plot can answer the

'why' question. However multioutput decision plots are more useful as it helps to visualize the impact of the features on all entities together. While this is more informative, not necessarily more intuitive for any lay user.

- Time efficiency – Time efficiency is calculated based on the unit of explanation used to generate the explanations for each method. It is worth noting that all the explanation visualizations presented in this study do not show all impacting features. Features that do not add much value are truncated for presentation. However, the calculation of this metric is performed on all features that had any non-zero influence. The rules for instance selection for this metrics calculation is discussed in Table 15 and custom weights for spelling and context related features are discussed in Table 16. Higher metrics value indicates better time efficiency of explanation.

Table 37 presents the time efficiency metrics explanation of 'LOC' prediction. It shows that Kernel SHAP explanations are more time-efficient in general. It is observed that Kernel SHAP explanation did not use any of the spelling and context related features for any of the samples used for this metrics calculation.

Table 37: Time efficiency calculation for prediction 'LOC' entity explanation

Entity: LOC	Accurate	Inaccurate
LIME-NER	-9.25	-9.71
Kernel SHAP	-3.88	-2.43

Table 38 presents the time efficiency metrics explanation of 'MISC' predictions. It shows that Kernel SHAP explanations are more time-efficient in general. It is observed that Kernel SHAP explanation used only a few of spelling and context related features (word_format, postag, word.isdigit, word.len) for explaining inaccurate 'MISC' predictions. It is also observed that time efficiency for explaining inaccurate 'MISC' prediction is much higher compared to that of accurate 'MISC' predictions.

Table 38: Time efficiency calculation for prediction 'MISC' entity explanation

Entity: MISC	Accurate	Inaccurate
LIME-NER	-8.17	-10
Kernel SHAP	-4.5	-7.7

- Comprehensibility – comprehensibility of explanation is understood by identity, stability and separability of explanation that is discussed in section 3.3.2. These are analysed by visualising word embedding after t-SNE transformation as discussed in section 4.5.1. Instance selection is discussed in Table 18 and visualization scenarios are discussed in Table 19.

  Figure 67 shows the words used in explaining accurate 'MISC' predictions by LIME-NER and Kernel SHAP. Red dots in the plot indicate words used only by Kernel SHAP, blue dots indicate words used only by LIME-NER and green dots indicate words used by both explanations. It is observed that there are broadly three clusters of words that are used for explanation in any method except the word 'PAD' that is visible towards the bottom of the plot. This word is only used by Kernel SHAP, but this is a non-word (it is the dummy word used to make variable length sentence fixed length). Therefore, this can be ignored.

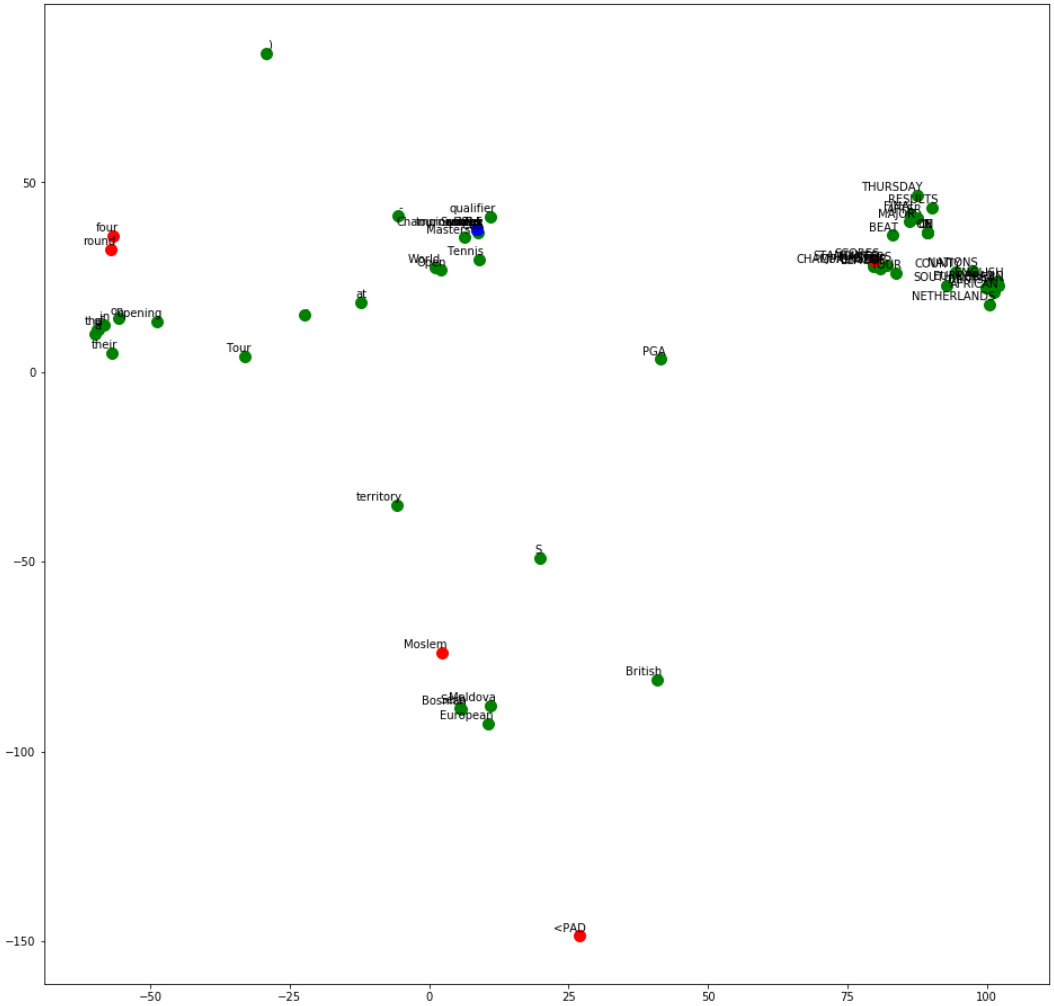

Figure 67: t-SNE visualisation for LIME-NER vs Kernel SHAP explanation for accurate MISC predictions

Figure 68 shows the words used in explaining accurate 'LOC' predictions by LIME-NER and Kernel SHAP. Red dots in the plot indicate words used only by Kernel SHAP, blue dots indicate words used only by LIME-NER and green dots indicate words used by both explanations. It is observed that the words are more scattered compared to the words used in 'MISC' prediction explanations. However, that is same for both LIME-NER and Kernel SHAP explanations.

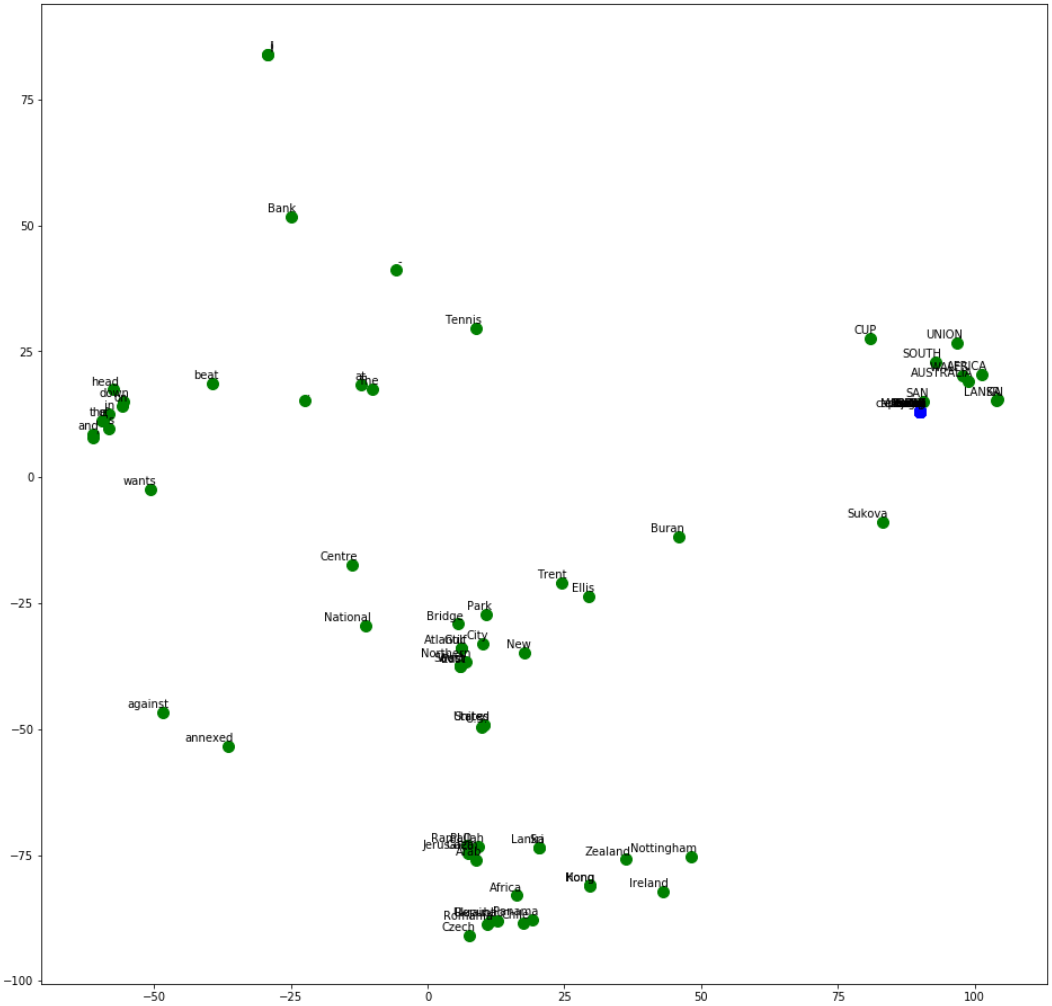

Figure 68: t-SNE visualisation for LIME-NER vs Kernel SHAP explanation for accurate LOC predictions

Figure 67 and Figure 68 are used to assess the identity and stability of explanations as they show the words used in explaining the same entity. These two figures suggest that explanation from LIME-NER and Kernel SHAP performs at par with respect to identity and stability.

Figure 69 shows the words used in explanation from LIME-NER for accurate 'LOC' and 'MISC' predictions. Red dots in the plot indicates words used for 'LOC', blue dots indicate words used for 'MISC' and green dots indicate words used for both explanations. It is observed that the number of common words used for 'LOC' and 'MISC' entity prediction is close to 15 in case of LIME-NER.

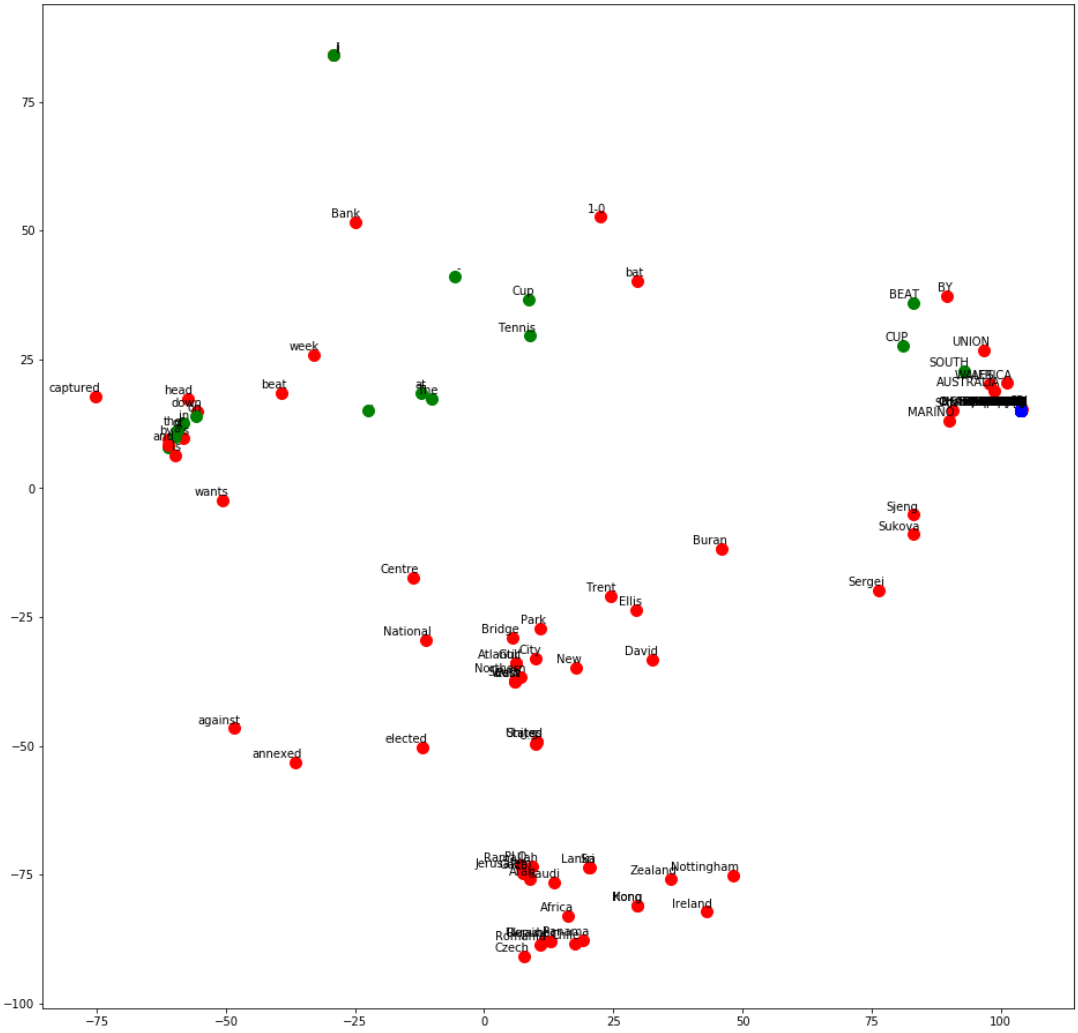

Figure 69: t-SNE visualisation for words used in LIME-NER explanation for accurate 'LOC' and 'MISC' predictions

Figure 70 shows the words used in explanation from Kernel SHAP for accurate 'LOC' and 'MISC' predictions. Red dots in the plot indicate words used for 'LOC', blue dots indicate words used for 'MISC' and green dots indicate words used for both explanations. It is observed that the number of common words used for 'LOC' and 'MISC' entity prediction explanation is close to 10 in case of Kernel SHAP.

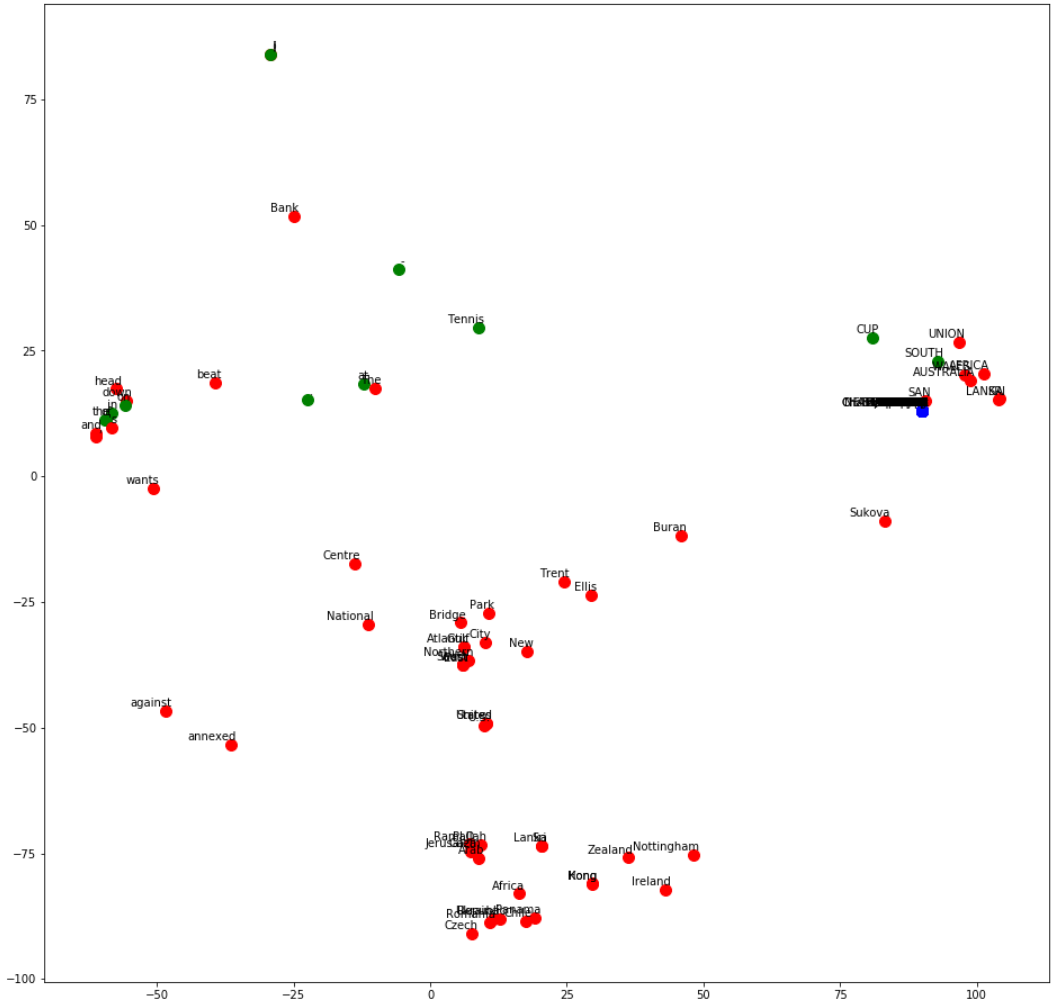

Figure 70: t-SNE visualisation for words used in Kernel SHAP explanation for accurate 'LOC' and 'MISC' predictions

As Figure 69 and Figure 70 shows the words used for explaining prediction of different entities by each method, less number of common words in explanation of two different entities suggest better separability. I.e. Kernel SHAP performs marginally better in separability compared to LIME-NER

5.5.2 Quantitative evaluation

This section will discuss the quantitative evaluation metrics as designed in 4.5.2. The area over the perturbation curve (AOPC) is calculated for 'LOC' and 'MISC' entity tag for both accurate and inaccurate predictions as designed in Table 21. Instances for explanation are selected as per the rules Table 20. Figure 71 shows the metrics generated for all the scenarios.

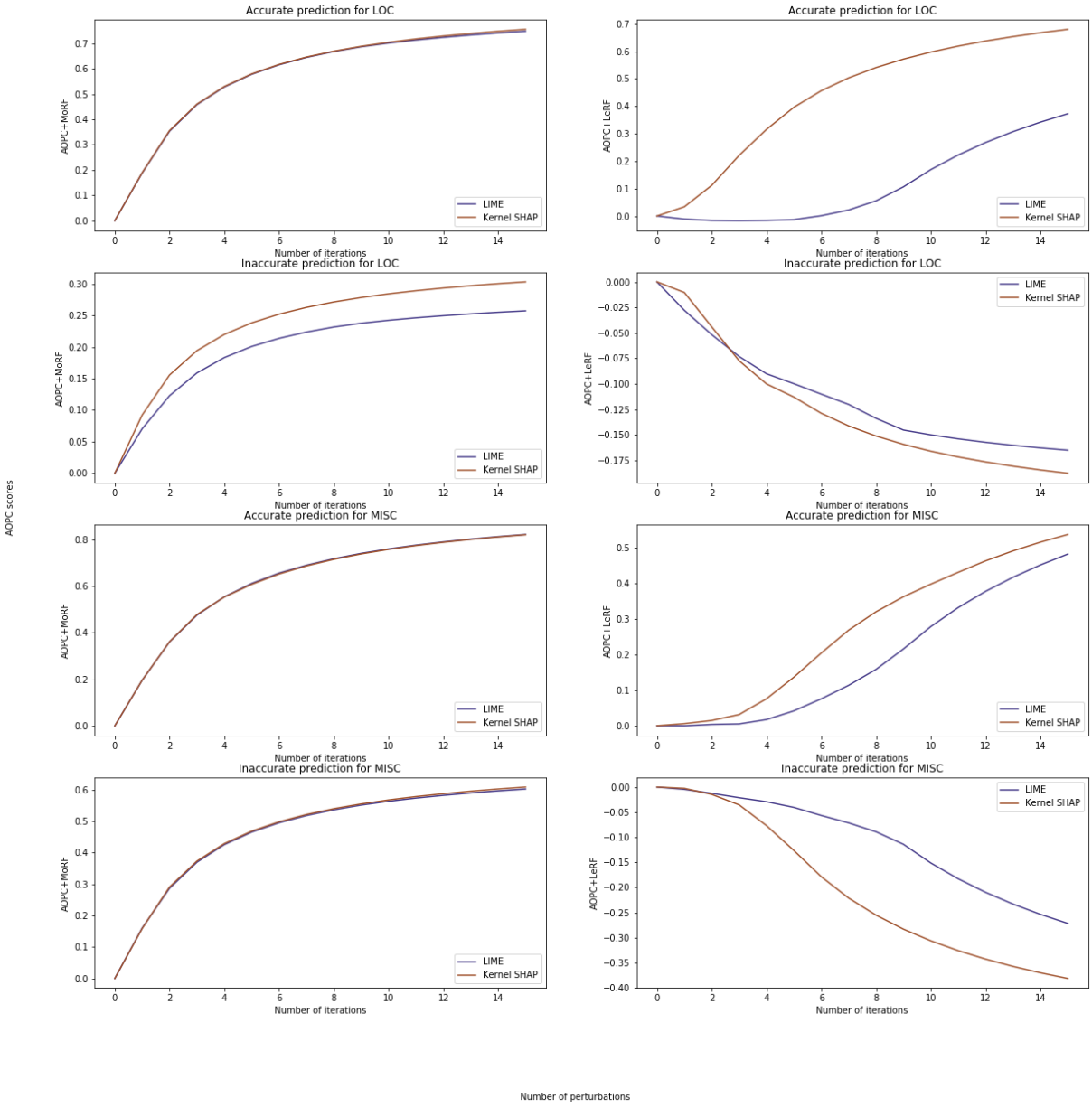

Figure 71: AOPC metrics

Analysis of the AOPC metrics for 'LOC' entity is summarised in Table 39 and for 'MISC' entity in Table 40.

Table 39: AOPC metric for 'LOC' prediction

Entity: LOC	AOPC+MoRF	AOPC+LeRF
Accurate prediction	Plot at the first row, first column in Figure 71 – steepness for both LIME-NER and Kernel SHAP curves are same.	Plot at the first row, second column in Figure 71 – the steepness of LIME-NER curve is lesser than the steepness of Kernel SHAP curve. This indicates LIME-NER explanation has redundant features. This observation is also supported by the lower value of time efficiently metric for LIME-NER as shown in Table 37. Additionally, the variance in the AOPC curve is high for Kernel SHAP compared to LIME-NER which indicates Kernel SHAP is a better method of explanation.
Inaccurate Prediction	Plot at the second row, first column in Figure 71 – the curve for Kernel SHAP is steeper than the curve for LIME-NER indicating Kernel SHAP can capture right order of influencing features for ground truth class even in case of wrong predictions. The variance of Kernel SHAP curve is higher as well	Plot at the second row, second column in Figure 71 – initial steepness of the curve for LIME-NER is marginally higher than Kernel SHAP. Along the curve, steepness for Kernel SHAP curve increased over LIME-NER. The variance of Kernel SHAP curve is higher than LIME-NER curve. This indicates removing the least important features in Kernel SHAP explanation for ground truth entity class will result in an increase of ground truth prediction probability in a much higher rate.

Table 40: AOPC metric for 'MISC' prediction

Entity: MISC	AOPC+MoRF	AOPC+LeRF
Accurate prediction	Plot at the third row, first column in Figure 71 – steepness for both LIME-NER and Kernel SHAP curves are same.	Plot at the third row, second column in Figure 71 – the steepness of LIME-NER curve is lesser than the steepness of Kernel SHAP curve. This indicates LIME-NER explanation has redundant features. This observation is also supported by the lower value of time efficiently metric for LIME-NER as shown in Table 38. Additionally, the variance in the AOPC curve is high for Kernel SHAP compared to LIME-NER which indicates Kernel SHAP is a better method of explanation.
Inaccurate Prediction	Plot at the fourth row, first column in Figure 71 – steepness for both LIME-NER and Kernel SHAP curves are same.	Plot at the fourth row, second column in Figure 71 – the steepness of the curve for Kernel SHAP is higher than the curve for LIME-NER. The variance for the Kernel SHAP curve is higher as well. This indicates removing the least important features in Kernel SHAP explanation for ground truth entity class will result in an increase of ground truth prediction probability in a much higher rate.

Analysis of AOPC metrics suggests that Kernel SHAP is better than LIME-NER in capturing the most important features in predicting entity tags while can limit the number of features that are required for explaining a prediction.

5.5.3 Insights for NER model improvement

This subsection will discuss the observations from explanations generated from both LIME-NER and Kernel SHAP. There are a couple of observations that can help to improve the NER model.

It is noted that Kernel SHAP has not used spelling and context related features as top influencers in explaining any of the accurate explanations that are used for visualization or metrics calculation. It is observed that Kernel SHAP used spelling and context related features in explaining a few inaccurate 'MISC' entity tag predictions. These insights were unique to Kernel SHAP explanations as LIME-NER explanations use words but not raw features for explaining a prediction. This observation inspired a few modifications in the initial NER model that is discussed in the next subsection

Analysis of Kernel SHAP explanations, especially the explanations for inaccurate predictions (section 5.4.3, 5.4.4, 5.4.7, and 5.4.8) show very high dependency on the words known to the pre-built GloVe word embedding. For example, in section 5.4.3, the word 'Washington' in the sentence 'I think he got a little tight at a couple of moments , said Washington .' predicted as 'LOC' entity inaccurately. The right entity tag for this word is 'PER'. The Kernel SHAP multioutput decision plot in Figure 42 shows that the context word 'said' has a more positive influence on the prediction of 'PER' entity tag than the prediction of 'LOC' entity tag for the word 'Washington'. However, the entity word 'Washington' has a very high influence on the prediction of 'LOC' entity. It indicates that the influence from the context words is overridden by the influence of the entity word. In another example in section 5.4.8, the word 'Missing' in the sentence 'Missing German girl found alive at Dutch campsite .' predicted as 'MISC' entity tag inaccurately. The correct entity tag for this word is 'O'. In this case, the entity word is not part of the GloVe word embedding, thus unknown to training. The word 'German' (this is present in GloVe embedding) that is appearing immediately after the entity word has a very high influence in predicting 'MISC' entity tag for the word 'Missing'. The NER prediction for the word 'German' is 'MISC' too. As per the multioutput decision plot in Figure 54, the presence of the word 'German' has overridden influences from its subsequent words. This high dependency on entity words and words in the immediate surrounding that are present in pre-trained GloVe embedding need further analysis to understand whether the NER model is able to learn right patterns from all context words. Though this insight can be theoretically achieved from LIME NER explanations (by visualizing the impact of words on each pair of

entity tags), Kernel SHAP multioutput decision plot helps in understanding the impact of each feature on all entity tags together. Additionally, the feature enrichment step in section 4.4.3 helps to identify the words that are outside of training vocabulary in the Kernel SHAP visualisations easily.

5.5.4 Performance metrics for modified BI-LSTM-CRF models

This section discusses the impact of the modifications on the NER model architecture that are designed in section 4.5.3 based on the observations from Kernel SHAP generated explanations in the previous subsections.

Table 41 compares the sequence F1 score on test-b data set for initial NER model, NER model with spelling and context related features for single word (i.e. no context word) and NER model without spelling and context related features.

Table 41: Sequence F1 score comparison between initial and modified NER models for the test-b dataset

Entity	Sequence F1 score		
	Initial NER model	NER model with spelling and context related features for single word only	NER model without spelling and context related features
LOC	0.84	0.84	0.85
PER	0.73	0.74	0.74
ORG	0.75	0.74	0.76
MISC	0.60	0.60	0.58
Macro average	0.76	0.76	0.76

Table 42 compares the sequence F1 score on test-a data set for initial NER model, NER model with spelling and context related features for only single surrounding words and NER model without spelling and context related features.

Table 42: Sequence F1 score comparison between initial and modified NER models for the test-a dataset

Entity	Sequence F1 score		
	Initial NER model	NER model with spelling and context related features for single word only	NER model without spelling and context related features
LOC	0.89	0.89	0.90
PER	0.78	0.81	0.82
ORG	0.78	0.79	0.77
MISC	0.76	0.75	0.77
Macro average	0.82	0.82	0.83

While the comparison of sequence F1 scores for all three version of models on both the data sets does not show any improvement post modification, it does not show any reduction in performance metrics either. This indicates that similar performance in NER prediction on these datasets is achievable with lesser complexity of the model. Table 22 shows how the trainable parameters in CRF layer in the NER model are reduced drastically due to the designed modifications. Comparison of prediction time for each of the model in Table 43 proves the same point.

Table 43: Prediction time comparison between initial and modified NER models

	Prediction from initial NER model	Prediction from NER model with spelling and context related features for single word only	Prediction from NER model without spelling and context related features
Number of sentences predicted	5389	5389	5389
Total time taken	24 seconds	15 seconds	10 seconds
Average time of prediction for each sentence	4.5 milliseconds	2.8 milliseconds	1.9 milliseconds

5.6 Summary

This section first presents the NER model performance on test-a data set using the sequence F1 score to establish that reasonable NER model is built for generating explanations. Then the section presents explanations generated using both LIME-NER and Kernel SHAP with interpretation for all visualizations. It also provides a comparison between explanations generated using these two methods.

Post visualization of explanations, the study compares the explanation from LIME-NER and Kernel SHAP using the qualitative and quantitative evaluation metrics designed in section 4.5.1 and 4.5.2 respectively. While the comparison of most of these metrics suggests that LIME-NER and Kernel SHAP performs at par in explaining accurate and inaccurate predictions, one qualitative evaluation metrics, time efficiency, and the quantitative metrics AOPC indicates that use of Kernel SHAP has marginal advantages.

This claim is strengthened in the discussion of model improvement insights from Kernel SHAP explanations. LIME-NER is not designed to provide explanations using raw features. Kernel SHAP's explanation using raw features could help in making modifications in the initial NER model. This section also discusses the observation around NER model's high dependency on entity words and words in the immediate surrounding that are present in pre-trained GloVe embedding. This provides further opportunity to analyse the NER model outcome in this light.

Finally, the NER model architecture is modified in two steps based on the insights from Kernel SHAP explanation. The performance of initial NER model and two modified models are compared using sequence F1 score on test-a and test-b data sets.

CHAPTER 6: CONCLUSIONS AND RECOMMENDATIONS

6.1 Introduction

This section will discuss how this study performed in meeting the aims and objectives from section 1.3. The section will also discuss the answers gathered as part of this study for the research questions in section 1.4.

This section will summarise the contribution of this study in the field of AI explainability. Finally, the section will discuss the limitations of this study and the areas that can be extended in future studies.

6.2 Discussion and Conclusion

An extensive literature review of AI explainability suggested lack of study in the domain of Named Entity Recognition that is a sequence tagging task in NLP. Concretely, there is only one study available for local explanation of NER predictions using post-hoc AI explainability method. That study used an improvised version of LIME, naming LIME-NER, for explaining NER predictions locally. This inspired the aim of this study and the primary research question, i.e. to generate a local explanation for NER predictions using another recent and state-of-the-art post-hoc AI explainability technique named Kernel SHAP.

Answering to the first research question - '*Is Kernel SHAP an effective method to explain Named Entity Recognition model built using BI-LSTM-CRF architecture*?', the study clearly demonstrated that Kernel SHAP can be used to generate the local explanation for NER predictions. Compared to the use of LIME for NER explanations, use of Kernel SHAP needed more post-processing of NER prediction outcome, because unlike LIME, Kernel SHAP method does not have any special treatment for textual data. However, this created an opportunity for the study to enrich the raw features for NER prediction with additional information that is easy to comprehend by the user. Additionally, this made it possible for the study to analyse the influence of non-text features on the prediction that was used as input in the BI-LSTM-CRF based NER model. It was not possible for LIME to visualise the influence of non-text features on NER prediction.

Literature review on AI explainability highlighted a few characteristics of a good explanation for the human audience. However, it also suggested a lack of consistency in evaluation metrics for generated explanations. The second objective and related research question, '*Are the explanations generated using Kernel SHAP for Named Entity Recognition model easy to understand for layman users?*', designed to find answers for the same. The study evaluated the explanations generated from LIME-NER and Kernel SHAP using three different qualitative evaluation methods that focused on the traits of good explanation from the phycological aspect.

Accuracy of an explanation is determined by the ability to answer contrastive 'why' questions. This is a necessary characteristic of an explanation for a human to consume. In the context of NER prediction, explanation is expected to answer why entity 'A' is predicted for a set of words instead of entity 'B' where both are possible. The visual comparison of LIME-NER and Kernel SHAP explanation for scenarios where the prediction probabilities of two entity classes were very close revealed that both the methods can answer the 'why' question. While the explanations from LIME-NER was more intuitive due to its ability to present the explanation in the form of highlighted words in the original sentence, the multioutput decision plot from Kernel SHAP was more informative due to its ability to display influence of each feature on all the possible entity tags together.

Time efficiency of an explanation indicates the time required for a human to go through an explanation. This is directly proportional to the number of units, i.e. features or words used to present the explanation. As Kernel SHAP used both word and spelling and context related features to present explanations, a novel technique was applied to calculate time efficiency. This used a variable weighting scheme for spelling and context related features depending on the complexity of the feature (i.e. more complex features would take more time to understand). Comparison of calculated time efficiency metrics for explanations from LIME-NER and Kernel SHAP indicated that Kernel SHAP used fewer features for delivering explanation and therefore more time-efficient.

Comprehensibility of an explanation varies depending on the audience of explanation and can be truly assessed in '*Human level evaluation'*. However, the current setup of this study

used only '*Function level evaluation*' for evaluating this characterises, i.e. using proxy tasks like identity, stability, and separability. Identity and stability are understood by the similarity of words used to explain multiple instances of the same entity. This study used t-SNE representation of word vectors from trained NER model for top 5 words appeared in LIME-NER and Kernel SHAP explanation. The comparison suggested that both explainability methods used a similar set of words in explanation and therefore they were at par with respect to identity and stability. Similarly, separability was understood by the dissimilarity of words used in explaining multiple instances of different entities. Visualisation of t-SNE representation of such words indicated Kernel SHAP used less similar words in explaining two different types of entities. Therefore, Kernel SHAP performed better with respect to separability.

The study also used one quantitative evaluation metric for explanations that have been used in a couple of earlier studies. The area over perturbation curve (AOPC) with MoRF (most relevant first) and LeRF (least relevant first) perturbation methods were experimented to understand whether explanation methods assigned right order of priority to features that are used for presenting the explanation. The right order of priority for features was understood by observing the pattern of increasing or decreasing NER prediction probabilities after each perturbation. Comparison of AOPC metrics clearly highlighted that Kernel SHAP has performed better than LIME-NER in most of the aspects that AOPC is designed to assess. Kernel SHAP demonstrated the ability to capture the right order of importance for the features while could limit the number of features that were required for explaining a prediction.

The final objective of the study was to inspect the explanations generated from LIME-NER and Kernel SHAP to find insights that can be used to improve the NER model. Kernel SHAP explanations proved to be better in this aspect compared to LIME-NER. Kernel SHAP's explanation using raw features helped to identify that the spelling and context related features were not necessary for building the NER model as none of its explanation of correct predictions used those features. The explanation of LIME-NER was not able to provide this insight as LIME-NER was designed to use word features which abstract raw features in word level. Modification of the NER model as per the insight did not increase or decrease the sequence F1 score that was used to evaluate NER models. But the

complexity of the feature creation and model building reduced drastically and that resulted in a reduction in prediction time of the NER model by more than a half.

Analysis of explanations of inaccurate predictions by Kernel SHAP method highlighted that the predictions from the NER model might not have enough influence from the words that are not in the immediate surrounding of the entity words. This seemed to be a drawback of the NER model that poses an opportunity for future improvement. However, for the objective of extracting model improvement insights from the explanations and the research question - '*Can the explanations generated using Kernel SHAP for Named Entity Recognition model provide insights related to shortcomings of the model?*', Kernel SHAP proved to do better compared to LIME-NER as this observation from multioutput decision plot was more direct.

After visual inspection of the explanations and analysis of evaluation results on explanations generated from LIME-NER and Kernel SHAP, the study concludes that Kernel SHAP has performed better in many of qualitative and quantitative evaluation metrics set up in the study. However, compared to LIME-NER, Kernel SHAP lacks in the presentation of explanation for text content. As originally proposed LIME had a special treatment for text data, its implementation comes with the ability to explain prediction on a text by highlighting words on the sentence used in prediction. In the case of Kernel SHAP, it is required to use the visualisations that were originally designed for structured data. This can be easily overcome by implementing a rich visualisation for text contents to present the explanation of Kernel SHAP and that will have the opportunity to design this specifically for NER predictions.

6.3 Contribution to knowledge

The study performed an extensive literature review in the domain of AI explainability covering what and why of AI explainability along with what human expects from explainability that AI systems should deliver. The literature review discussed the AI explainability work done in NLP tasks including NER with the focus on post-hoc local explainability techniques. The study did not find any existing literature that summarises

the AI explainability methods attempted on NLP tasks. This study is able to present this unique knowledge to the readers of the document.

Use of Kernel SHAP for NER explanation being the primary focus of this study, the design for the same is novel to this study and is a significant contribution to knowledge in the domain of AI explainability on NER tasks. The detailed discussion of NER prediction post-processing steps including feature enrichment, setup of Kernel SHAP specific to NER explanations, choice of visualisation for presenting explanations can be used as a basis for any future work in this domain.

The concepts of the qualitative evaluation metrics used in this study for evaluation of explanations are available in the existing literature. However, the formulation of time efficiency metric is a novel idea proposed in this study. This evaluation metric can be used to evaluate the time efficiency of any explanation and especially useful when units of explanation are heterogeneous in nature. I.e. in the context of Kernel SHAP, the explaining feature can be either words or morphological features extracted from the words. While calculating the time efficiency of explanation, the formulation of the metric can use different weights for different type of features based on ease of understanding.

6.4 Future Recommendations

Kernel SHAP provides a methodology to generate global explanation of a model by accepting multiple data instances together and then generating the SHAP values for each instance individually. Essentially SHAP values are a three-dimensional matrix where the dimensions are number of instances for explanation, number of target classes and number of features. The global explanations from Kernel SHAP are particularly useful as both global and local explanation are generated using the SHAP value and therefore are always consistent.

Generation of Kernel SHAP global explanation needs the predict function to be same for all instances. The current design of generating a local explanation for NER using Kernel SHAP does not support the use of same prediction function for multiple instances. In the current setup of the study, it is necessary to instantiate the NER prediction function wrapper

(that is discussed in section 4.4.3) with the position of the words that are part of the entity that needs explanation. This step ensures that post NER prediction on a sentence, only the prediction probabilities for those words are emitted to Kernel SHAP to understand the influence of coalition vector perturbation. This limitation in the study prevents the generation of global explanations on NER prediction. This can be an area of future exploration to overcome the limitation so that Kernel SHAP can be used for both local and global explanation of NER task.

Finally, visual assessment of Kernel SHAP explanations suggests that current visualizations offered by Kernel SHAP are not intuitive for textual contents. While LIME offers better text content specific visualization, that does not seem to be apt in NER set up. The visualisation offered by LIME highlights the positive and negative influencing words in the sentence. This visual design is suitable for text classification task on the sentence. However, this cannot highlight the entity words in the sentence that are subject of explanation. This provides an improvement opportunity in explanation visualization that can present all important and necessary elements of a NER prediction explanation in a single view.

REFERENCES

[1] Abdul, A., Vermeulen, J., Wang, D., Lim, B.Y. and Kankanhalli, M., (2018) Trends and trajectories for explainable, accountable and intelligible systems: An HCI research agenda. In: *Conference on Human Factors in Computing Systems - Proceedings*. Association for Computing Machinery.

[2] Alvarez-Melis, D. and Jaakkola, T.S., (2017) *A causal framework for explaining the predictions of black-box sequence-to-sequence models*. [online] *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*. Association for Computational Linguistics (ACL). Available at: http://arxiv.org/abs/1707.01943 [Accessed 2 May 2020].

[3] Ancona, M., Ceolini, E., Öztireli, C. and Gross, M., (2017) Towards better understanding of gradient-based attribution methods for Deep Neural Networks. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*. [online] Available at: http://arxiv.org/abs/1711.06104 [Accessed 8 Feb. 2020].

[4] Andrews, R., Diederich, J. and Tickle, A.B., (1995) Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, 86, pp.373–389.

[5] Angwin, J. and Larson, J., (2016) *Machine Bias — ProPublica*. [online] Available at: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing [Accessed 25 Apr. 2020].

[6] Anon (2020) *AIC vs. BIC – The Methodology Center*. [online] Available at: https://www.methodology.psu.edu/resources/AIC-vs-BIC/ [Accessed 30 Jul. 2020].

[7] Anon (2020) *Explainable Artificial Intelligence*. [online] Available at: https://www.darpa.mil/program/explainable-artificial-intelligence [Accessed 24 Apr. 2020].

[8] Anon (2020) *Explainers — SHAP latest documentation*. [online] Available at: https://shap.readthedocs.io/en/latest/ [Accessed 11 May 2020].

[9] Anon (2020) *General Data Protection Regulation - Wikipedia*. [online] Available at: https://en.wikipedia.org/wiki/General_Data_Protection_Regulation [Accessed 24 Apr. 2020].

[10] Anon (2020) *GloVe: Global Vectors for Word Representation*. [online] Available at: https://nlp.stanford.edu/projects/glove/ [Accessed 8 Mar. 2020].

[11] Anon (2020) *Language-Independent Named Entity Recognition (II)*. [online] Available at: https://www.clips.uantwerpen.be/conll2003/ner/ [Accessed 14 Mar. 2020].

[12] Anon (2020) *Myth busters*. [online] Available at: https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public/myth-busters [Accessed 1 May 2020].

[13] Anon (2020) *Understanding LSTM Networks -- colah's blog*. [online] Available at: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ [Accessed 9 May 2020].

[14] Anon (n.d.) *Principles for Algorithmic Transparency and Accountability*.

[15] Arrieta, A.B., Díaz-Rodríguez, N., del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R. and Herrera, F., (2019) Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI. [online] October. Available at: http://arxiv.org/abs/1910.10045.

[16] Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R.R. and Samek, W., (2015) On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, [online] 107, pp.1–46. Available at: http://www.hfsp.org/, [Accessed 8 Feb. 2020].

[17] Belinkov, Y. and Glass, J., (2019) Analysis Methods in Neural Language Processing: A Survey. *Transactions of the Association for Computational Linguistics*, 7, pp.49–72.

[18] Bengio, Y., Simard, P. and Frasconi, P., (1994) Learning Long-Term Dependencies with Gradient Descent is Difficult. *IEEE Transactions on Neural Networks*, 52, pp.157–166.

[19] Bird, S., Klein, E. and Loper, E., (2009) *Natural Language Processing with Python*. [online] Available at: https://www.nltk.org/book/ [Accessed 30 May 2020].

[20] Caliskan, A., Bryson, J.J. and Narayanan, A., (2016) Semantics derived automatically from language corpora contain human-like biases. *Science*, [online] 3566334, pp.183–186. Available at: http://arxiv.org/abs/1608.07187 [Accessed 25 Apr. 2020].

[21] Carvalho, D. v., Pereira, E.M. and Cardoso, J.S., (2019) *Machine learning interpretability: A survey on methods and metrics*. *Electronics (Switzerland)*, .

[22] Collobert, R., Weston, J., Com, J., Karlen, M., Kavukcuoglu, K. and Kuksa, P., (2011) *Natural Language Processing (Almost) from Scratch*. *Journal of Machine Learning Research*, .

[23] Cramer, H., Evers, V., Ramlal, S., van Someren, M., Rutledge, L., Stash, N., Aroyo, L. and Wielinga, B., (2008) The effects of transparency on trust in and acceptance of a content-based art recommender. *User Modeling and User-Adapted Interaction*, 185, pp.455–496.

[24] Davis, R., Buchanan, B. and Shortliffe, E., (1977) Production Rules as a Representation for a Knowledge-Based Consultation Program. [online] Springer, New York, NY, pp.3–37. Available at: http://link.springer.com/10.1007/978-1-4612-5108-8_1 [Accessed 25 Apr. 2020].

[25] Doshi-Velez, F. and Kim, B., (2017) Towards A Rigorous Science of Interpretable Machine Learning. [online] Available at: http://arxiv.org/abs/1702.08608 [Accessed 9 Feb. 2020].

[26] Doshi-Velez, F. and Kim, B., (2018) Considerations for Evaluation and Generalization in Interpretable Machine Learning. pp.3–17.

[27] Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M. and Kagal, L., (2019) *Explaining Explanations: An Overview of Interpretability of Machine Learning*. *Proceedings - 2018 IEEE 5th International Conference on Data Science and Advanced Analytics, DSAA 2018*.

[28] Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Pedreschi, D. and Giannotti, F., (2018) A Survey Of Methods For Explaining Black Box Models. *ACM Computing Surveys*. [online] Available at: http://arxiv.org/abs/1802.01933 [Accessed 23 Apr. 2020].

[29] Herlocker, J.L., Konstan, J.A. and Riedl, J., (2000) *Explaining Collaborative Filtering Recommendations*. [online] Available at: http://www.grouplens.org/ [Accessed 25 Apr. 2020].

[30] Hironsan, (2019) *How to compute f1 score for named-entity recognition in Keras*. [online] Available at: https://towardsdatascience.com/how-to-compute-f1-score-for-named-entity-recognition-in-keras-6f28b31dccca [Accessed 14 Mar. 2020].

[31] Hochreiter, S. and Schmidhuber, J., (1997) Long Short-Term Memory. *Neural Computation*, [online] 98, pp.1735–1780. Available at: http://www.mitpressjournals.org/doi/10.1162/neco.1997.9.8.1735 [Accessed 9 May 2020].

[32] Honegger, M.R. and Blanc, S., (2018) *Shedding Light on Black Box Machine Learning Algorithms Development of an Axiomatic Framework to Assess the Quality of Methods that Explain Individual Predictions Master Thesis*. [online] Available at: www.kit.edu [Accessed 25 Apr. 2020].

[33] Huang, Z., Xu, W. and Yu, K., (2015) Bidirectional LSTM-CRF Models for Sequence Tagging. [online] Available at: http://arxiv.org/abs/1508.01991 [Accessed 29 Feb. 2020].

[34] Lafferty, J., Mccallum, A., Pereira, F.C.N. and Pereira, F., (2001) *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data Part of the Numerical Analysis and Scientific Computing Commons Recommended Citation "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data" Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*. [online] Available at: http://repository.upenn.edu/cis_papersPublisherURL:http://portal.acm.org/citation.cfm?id=655813PublisherURL:http://portal.acm.org/citation.cfm?id=655813ThisconferencepaperisavailableatScholarlyCommons:http://repository.upenn.edu/cis_papers/159 [Accessed 7 Mar. 2020].

[35] Lee, C., (2017) LSTM-CRF Models for named entity recognition. *IEICE Transactions on Information and Systems*, E100D4, pp.882–887.

[36] Letzter, R., (2016) *How algorithms can be racist - Business Insider*. [online] Available at: https://www.businessinsider.com/how-algorithms-can-be-racist-2016-4?IR=T [Accessed 25 Apr. 2020].

[37] Li, J., Chen, X., Hovy, E. and Jurafsky, D., (2016) Visualizing and understanding neural models in NLP. *2016 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference, [online] pp.681–691. Available at: https://arxiv.org/abs/1506.01066%0A.

[38] Li, S., (2019) *Explain NLP models with LIME & SHAP - Towards Data Science*. [online] Available at: https://towardsdatascience.com/explain-nlp-models-with-lime-shap-5c5a9f84d59b [Accessed 13 Mar. 2020].

[39] Liang, B., Li, H., Su, M., Bian, P., Li, X. and Shi, W., (2017) Deep Text Classification Can be Fooled. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. California: International Joint Conferences on Artificial Intelligence Organization.

[40] Lipton, P., (1990) Contrastive Explanation. *Royal Institute of Philosophy Supplement*, 27, pp.247–266.

[41] Lundberg, S. and Lee, S.-I., (2017) A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems*, [online] 2017-Decem, pp.4766–4775. Available at: http://arxiv.org/abs/1705.07874 [Accessed 8 Feb. 2020].

[42] Luo, L., Yang, Z., Yang, P., Zhang, Y., Wang, L., Lin, H. and Wang, J., (2018) An attention-based BiLSTM-CRF approach to document-level chemical named entity recognition. *Bioinformatics*, 348, pp.1381–1388.

[43] van der Maaten, L. and Hinton, G., (2008) *Visualizing Data using t-SNE*. *Journal of Machine Learning Research*, .

[44] Madigan, D., Mosurski, K. and Almond, R.G., (1997) Graphical explanation in belief networks. *Journal of Computational and Graphical Statistics*, 62, pp.160–181.

[45] Miller, T., (2019) *Explanation in artificial intelligence: Insights from the social sciences*. *Artificial Intelligence*, .

[46] Molnar, C., (2019) *Interpretable Machine Learning*. [online] Available at: https://christophm.github.io/interpretable-ml-book/ [Accessed 23 Feb. 2020].

[47] Nguyen, A., Yosinski, J. and Clune, J., (2014) Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, [online] 07-12-June-2015, pp.427–436. Available at: http://arxiv.org/abs/1412.1897 [Accessed 25 Apr. 2020].

[48] Nickerson, R.S., (1998) Confirmation bias: A ubiquitous phenomenon in many guises. *Review of General Psychology*, 22, pp.175–220.

[49] Núñez, H., Angulo, C. and Català, A., (2002) *Rule Extraction from Support Vector Machines*. Studies in Computational Intelligence. [online] Berlin, Heidelberg: Springer Berlin Heidelberg. Available at: http://link.springer.com/10.1007/978-3-540-75390-2 [Accessed 25 Apr. 2020].

[50] Olah, C., Mordvintsev, A. and Schubert, L., (2017) Feature Visualization. *Distill*, [online] 211, p.e7. Available at: https://distill.pub/2017/feature-visualization [Accessed 27 Apr. 2020].

[51] Ribeiro, M.T., Singh, S. and Guestrin, C., (2016) "Why Should I Trust You?" Explaining the Predictions of Any Classifier. [online] Available at: http://dx.doi.org/10.1145/2939672.2939778 [Accessed 8 Feb. 2020].

[52] Ribeiro, M.T., Singh, S. and Guestrin, C., (2018) *Anchors: High-Precision Model-Agnostic Explanations*. [online] Available at: www.aaai.org [Accessed 8 Feb. 2020].

[53] Ribera, M. and Lapedriza, A., (2019) *Can we do better explanations? A proposal of User-Centered Explainable AI*.

[54] Robinson, S. and Yufeng, (2019) *Predicting Stack Overflow Tags with Google's Cloud AI - Stack Overflow Blog*. [online] Google Next 2019 conference. Available at: https://stackoverflow.blog/2019/05/06/predicting-stack-overflow-tags-with-googles-cloud-ai/ [Accessed 13 Mar. 2020].

[55] Ruping, S., (2006) *Learning Interpretable Models*.

[56] Samek, W., Binder, A., Montavon, G., Lapuschkin, S. and Müller, K.R., (2017a) Evaluating the visualization of what a deep neural network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, 2811, pp.2660–2673.

[57] Samek, W., Wiegand, T. and Müller, K.-R., (2017b) *EXPLAINABLE ARTIFICIAL INTELLIGENCE: UNDERSTANDING, VISUALIZING AND INTERPRETING DEEP LEARNING MODELS*. [online] Available at: http://arxiv.org/abs/1708.08296 [Accessed 8 Feb. 2020].

[58] Shrikumar, A., Greenside, P. and Kundaje, A., (2017) Learning Important Features Through Propagating Activation Differences. *34th International Conference on Machine Learning, ICML 2017*, [online] 7, pp.4844–4866. Available at: http://arxiv.org/abs/1704.02685 [Accessed 8 Feb. 2020].

[59] Simonyan, K., Vedaldi, A. and Zisserman, A., (2013) Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *2nd International Conference on Learning Representations, ICLR 2014 - Workshop Track Proceedings*. [online] Available at: http://arxiv.org/abs/1312.6034 [Accessed 8 Feb. 2020].

[60] Sundararajan, M., Taly, A. and Yan, Q., (2017) Axiomatic Attribution for Deep Networks. *34th International Conference on Machine Learning, ICML 2017*, [online] 7, pp.5109–5118. Available at: http://arxiv.org/abs/1703.01365 [Accessed 8 Feb. 2020].

[61] Swartout, W.R., (1983) XPLAIN: a system for creating and explaining expert consulting programs. *Artificial Intelligence*, 213, pp.285–325.

[62] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. and Fergus, R., (2014) *Intriguing properties of neural networks*.

[63] Vanni, L., Ducoffe, M., Mayaffre, D., Precioso, F., Longrée, D., Elango, V., Santos Buitrago, N., Gonzales Huesca, J., Galdo, L., Aguilar, C., Text, al, Vanni, L., Ducoffe, M., Mayaffre, D., Precioso, F., Longrée, D., Elango, V., Santos, N., Gonzalez, J., Galdo, L. and Aguilar, C., (2018) *Text Deconvolution Saliency (TDS) : a deep tool box for linguistic analysis Deconvolution Saliency (TDS) : a deep tool box for linguistic analysis Text Deconvolution Saliency (TDS) : a deep tool box for linguistic analysis*. [online] Available at: https://hal.archives-ouvertes.fr/hal-01804310 [Accessed 13 Mar. 2020].

[64] Villarroya, S.H., (2018) *Interpretability in sequence tagging models for Named Entity Recognition*.

[65] Volpato, R., (2019) *Riccardo Volpato | NLP meets XAI: Top 5 Trends in NLP Explainability*. [online] Available at: https://volpato.io/blog/2019/nlp-meets-xai/ [Accessed 2 May 2020].

[66] Wachter, S., Mittelstadt, B. and Russell, C., (2017) Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. *SSRN Electronic Journal*. [online] Available at: http://arxiv.org/abs/1711.00399 [Accessed 27 Apr. 2020].

[67] Wei, H., Gao, M., Zhou, A., Chen, F., Qu, W., Wang, C. and Lu, M., (2019) Named Entity Recognition from Biomedical Texts Using a Fusion Attention-Based BiLSTM-CRF. *IEEE Access*, 7, pp.73627–73636.

[68] Xie, Y., Le, L., Zhou, Y. and Raghavan, V. v., (2018) Explainable Deep Learning for Natural Language Processing. In: *Handbook of Statistics*. [online] pp.317–328. Available at: http://kth.diva-portal.org/smash/get/diva2:1335846/FULLTEXT01.pdf [Accessed 27 Jan. 2020].

[69] Zeiler, M.D. and Fergus, R., (2014) Visualizing and understanding convolutional networks. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer Verlag, pp.818–833.

APPENDIX A: RESEARCH PROPOSAL

Abstract

Named Entity Recognition (NER) is a common Natural Language Processing task for extracting important information from documents. This technique is used very frequently in an organisational setup where the requirement is to extract key information from the digitised documents so that the information can be fed to further downstream applications. Organisations expect both accuracy and trust in the information extraction process before they can use this process in practice. In recent days, Deep Neural Network (DNN) based architectures can achieve state-of-the-art accuracy in NER tasks. However, high accuracy alone is not enough to build trust amongst users, especially when black-box models like DNN is used. With increasing use of deep learning in the fields like image, text etc., various model specific and model agnostic ways of AI explainability techniques have been proposed in recent years. Earlier studies have shown how one deep learning specific explanation technique - Layer-wise Relevance Propagation and one model agnostic explanation technique – Local Interpretable Model-Agnostic Explanations (LIME) can be adapted to explain NER models. This study will demonstrate how a more recent technique named Kernel SHAP, proposed based on LIME and classical Shapely Values from game theory, can be used to generate explanation for a DNN based NER model. The study aims to generate explanations that can be useful for users with sufficient language understanding but no formal ML knowledge. The study also aims to analyse how insights gathered from explanations of wrong predictions can be used to improve the NER model.

1. Background

In the domain of Artificial Intelligence (AI), Natural Language Processing (NLP) is the field that focuses on how computers can process and interpret natural language data. Due to complexity of any natural language, study of natural language is subdivided in various sub-fields (Outline of natural language processing - Wikipedia, 2020).

Named entity recognition (NER) is a classic NLP task which comes under Information Extraction (IE) sub-field. This is a very common but non-trivial task in retrieving information from documents. NER is a sequence tagging problem where each word in a sentence (i.e. sequence of words) are marked whether that is part of an entity or otherwise. What makes this task more challenging is that only a small subset of words in a sentence is part of an entity. This makes modelling an NER task tricky as overall performance metric aggregated over all entities does not represent the performance of the individual entity.

One of the widely used probabilistic models for NER has been Conditional Random Field (CRF) as proposed by (Lafferty et al., 2001). It is possible to interpret a CRF model by looking at the feature weights calculated by CRF algorithm. Additionally, the features are carefully curated by Machine Learning (ML) practitioners for CRF, therefore interpreting the features does not need additional effort. This is almost exactly the same as logistic regression where coefficients of features represent their importance. Training a CRF means to compute the optimal weights for each feature given the observed sequence of words.

Recently, Bi-directional Long Short-Term Memory and Conditional Random Field (BI-LSTM-CRF) architecture, as proposed by (Huang et al., 2015), has become very popular for sequence tagging task. This has proved to deliver better performance in entity extraction from text. However, any deep neural network model achieves its performance because of its complex architecture and ability to efficiently solve the huge parameter space and this makes deep learning algorithms black-box (not transparent by design) model (Arrieta et al., 2019). Therefore, NER model created using BI-LSTM-CRF does not offer same interpretability compared to a NER model created using CRF alone.

These issues can pose serious problems in utilizing the NER output from BI-LSTM-CRF based model in any practical use. For instance, in an organizational set up, where organizations have

migrated to digital document repository long back, it is often seen that extracting important information from the digitised documents is not easy task as documents have not been properly tagged or indexed to start with. When these organizations plan to use the information available in the documents for any further processing, Machine Learning practitioner resort to NER to extract important information. Both poor performance and lack of interpretability in entity extraction becomes hindrance in using the predictions for any downstream application as errors in information extraction potentially would lead to undesired consequences.

An extensive survey on eXplainable Artificial Intelligence (XAI) (Arrieta et al., 2019) stresses upon the point that performance metrics / accuracy of an ML model may be of interest for research studies, but the understanding of the ML model is necessary for practical use and further enhancement. The survey study states the following regarding consideration of interpretability while developing a ML model:

- Interpretability of a ML model makes it easier for ML practitioners to make the model accurate by removing unnecessary biases induced by training data.
- Interpretability in a model increases trustworthiness of the ML model and therefore increases chances of adoption of the ML model in any business process.

Explainability techniques for black-box models are referred to as post-hoc explainability methods. The primary objective for post-hoc explainability method is to make output of a non-interpretable model meaningful to human being. Please note that DNNs are not only non-interpretable modes; SVM and Tree based ensemble models are also considered to be back-models as they don’t provide transparency by design.

The survey (Arrieta et al., 2019) suggests various available approaches for post-hoc explainability. Below is the diagram from (Arrieta et al., 2019) that depicts all different post-hoc explainability approaches.

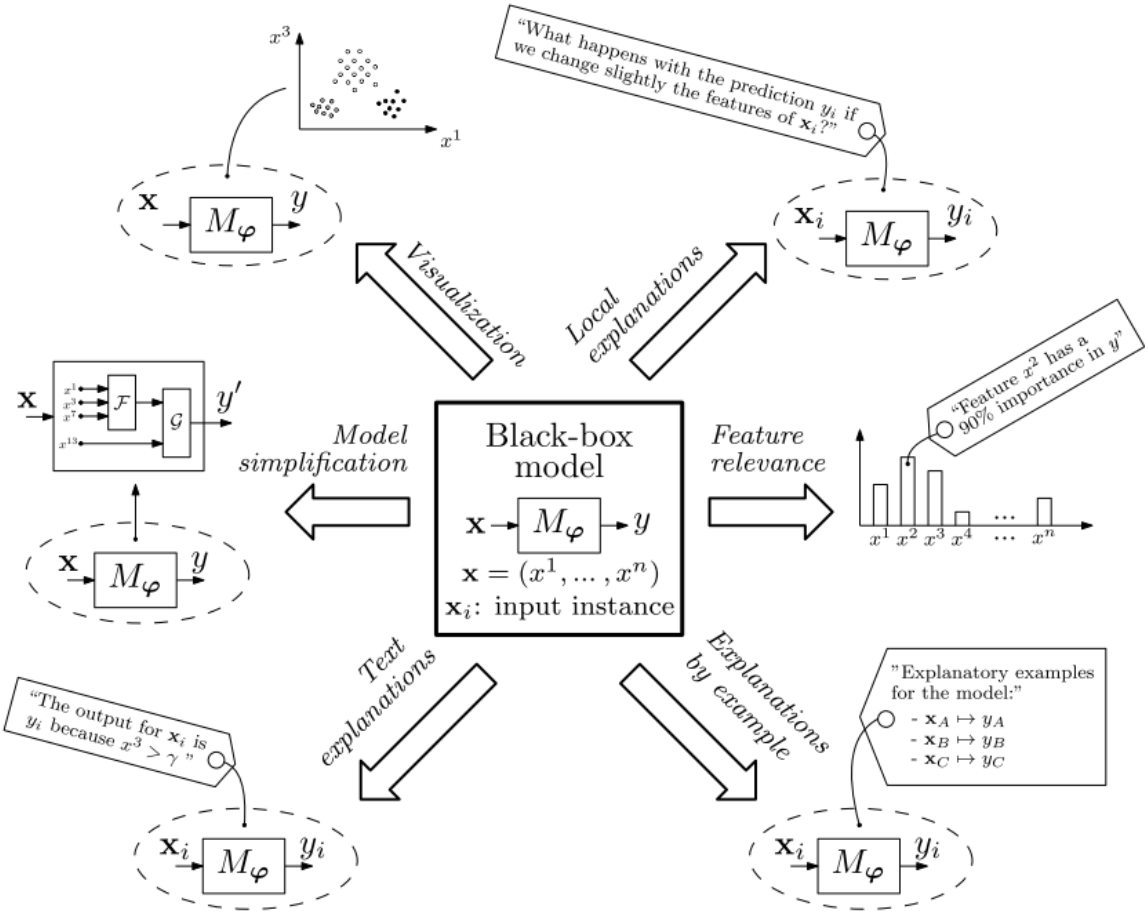

Figure 72: Post-hoc explainability approaches (Arrieta et al., 2019)

However, literature review indicates local explanations and feature relevance that are being used increasingly for explaining Deep Neural Networks (DNN). Local explanation of a model refers to interpretation of predictions on a single or set of observations. These methods, in general, try to approximate the model using an interpretable function locally to the set of observations. Majority of these techniques are model agnostic.

Feature relevance explainability techniques are suitable when global interpretability of a ML model is desired. This technique calculates importance score for each feature at input with respect to output. Feature relevance techniques can be both model agnostic and mode specific. This study will focus on couple of model agnostic local explainability techniques and one model specific feature relevance explainability techniques that are relevant to current study on interpretability of NER.

In literature review, it is noticed that AI explainability techniques were not studied on deep learning based NER models extensively. This provides an opportunity for the current research to study recently developed AI explainability techniques on deep learning based NER models.

The purpose the of the study will be to assess whether explanation generated for NER models can build trust for the predictions and help in improving model performance by generating insights from wrong predictions.

2 Problem Statement

This section will start with overview of related works in AI explainability and finally will explain the problem statement that the study will focus on. Noticeably, most of the explanation techniques were initially applied to image related tasks potentially due to easy visualization of input feature space; and later adapted for text. Therefore, this study will first discuss the work where the any particular explainability technique got proposed and then how that is adapted for text related task. Eventually, the study will discuss the techniques used for explainability on DNN based NER models.

2.1 AI Explainability and NLP

(Simonyan et al., 2013) used saliency scores to visualize image classification models, learnt using deep Convolutional Networks (ConvNets). This study proposed an understandable visualisation of ConvNet classification models that can be obtained using single backpropagation pass through a classification convolution network. This method computed an image specific class saliency map, highlighting the areas of the given image that discriminates the image from the other classes with respect to predicted class. This was model specific feature relevance explainability technique. Inspired by this work, (Li et al., 2016) proposed strategies for visualizing compositionality in neural models for NLP. This paper used t-SNE visualisation to understand negations, clause compositions in Stanford Sentiment Treebank data. Using Saliency maps it understood how much each input unit contributes to the final decision in a neural network set up for different architectures (RNN and Autoencoder) and different NLP tasks (Sentiment Classification and next word prediction). However, the task of NER was not part of that study.

(Zeiler and Fergus, 2014) proposed another model specific feature relevance explainability approach to visualise the activity within the model using Deconvolutional Network (deconvnet). Aim of deconvnet was to project the feature activations back to the input feature space. The methodology in the paper could demonstrate part of the image important for classification. This also shown how these visualisations can be used to identify problems with

the model and use them to get better models. (Vanni et al., 2018) proposed a new deconvolution strategy, Text Deconvolution Saliency (TDS), for convolution neural network based text classification inspired by deconvolution network proposed by (Zeiler and Fergus, 2014). This study also used the t-SNE based visualisation of compositionality from (Li et al., 2016). The aim of the study was to understand how linguistic structure is learned by ConvNet that helps to achieve high classification accuracy. The study used data sets from 3 different languages – English, French and Latin. The implementation in this paper had to improvise the deconvolution network proposed (Zeiler and Fergus, 2014) due difference in CNN architecture for Text and Images. The study demonstrated that the relevance score calculated using TDS can encode complex linguistic features like word co-occurrences and potentially other grammatical and syntactic structures. But this approach is not fit for explaining NER due to the difference in the nature of prediction task between NER and text classification.

(Bach et al., 2015) used Layer-wise Relevance Propagation (LRP) to understand how relevance score from the output layer can be distributed to the input layer. The aim of the paper was to understand the contribution of a single pixel of an image to the prediction made by a classifier in an image classification task. LRP redistribute the relevance score of activated nodes in the output layer to features in the input layer via the backpropagation operation. The relevance score refers to the value calculated by the activated node in the output layer. Inspired by the earlier study, (Samek et al., 2017b) compared LRP with sensitivity analysis (SA) for explaining the individual predictions of an AI model in terms of input features. This paper demonstrated explainability of a text classification for a word-embedding based convolutional neural network trained to classify text documents from the 20News group dataset. LRP identified words supporting the predicted class as well contradicting the predicted class. In this paper, researchers established a methodology to evaluate multiple explainability techniques. At every perturbation step, the most important words (according to SA or LRP score) are deleted by setting the corresponding input word embedding values to 0. The result confirmed quantitatively that LRP provided more interpretable results than SA. Though, LRP is model specific feature relevance explainability approach, this can be applied on any DNN model. (Xie et al., 2018) used LRP for interpreting BI-LSTM-CRF based NER model which is discussed in detail later in this section.

(Sundararajan et al., 2017) proposed a new technique named Integrated Gradients for deep networks combining the "*Implementation Invariance*" of gradients along with the "*Sensitivity"* from techniques like LRP. This technique attributed the prediction of a deep network to its inputs. This paper also highlighted desirable features of an attribution method using an axiomatic framework without which it is difficult to ascertain whether attribution method is affected by data, network or the attribution method itself. (Sundararajan et al., 2017) experimented their methodology for question classification task trained on WikiTableQuestions dataset. That study used integrated gradients to identify key phrases from questions that differentiates answer types (e.g. phrase 'how many' in question will predict answer type as Numeric, phrase 'which film' in question will predict answer type as String etc.). Similar to LRP (Bach et al., 2015), Integrated Gradients is model specific feature relevance explainability approach and can be applied on any DNN model. However, Integrated Gradients will have same disadvantaged as LRP (discussed later in this section) if used to interpret BI-LSTM-CRF based NER model.

(Shrikumar et al., 2017) proposed another model specific feature relevance explainability technique, DeepLIFT, which claimed to overcome former gradient based methods like Deconvolution Network, LRP, integrated gradients that could produce misleading importance scores in cases of zero or discontinuous gradients. DeepLIFT proposed computing importance scores based on the difference of the output from some 'reference' output in terms of differences of the inputs from the 'reference' inputs. This strategy was effective in case of zero and discontinuous gradients. DeepLIFT proposed to have different considerations for positive and negative attribution of features at nonlinear functions which is more effective compared to earlier approaches. It is observed that there could be couple of disadvantages of using DeepLIFT - (1) choosing appropriate reference input (especially for text related tasks) would need domain expertise (2) choosing appropriate rule ("*Linear*" vs "*Rescale*" vs "*RevealCancel*") for calculating contribution scores at nonlinear functions. However, (Lundberg and Lee, 2017) proposed new technique, DeepSHAP, based on DeepLIFT and Shapely Values (based on game theory) that can address the inconvenience of choosing rule heuristically. DeepSHAP is discussed later in the section along with other SHAP techniques. Literature review done so far could not discover an instance where DeepLIFT has been adapted for any text related tasks. The original study used one image classification (MNIST data set) and one simulated Genomics classification data set to compare importance scoring.

All the explainability techniques mentioned above are model specific techniques where the strategy was to use internals of the model to arrive at interpretability. (Ribeiro et al., 2016) proposed Local Interpretable Model-Agnostic Explanations (LIME) which was a model agnostic approach to explain the predictions in an interpretable manner. The main idea of LIME was to learn a new locally interpretable (potentially linear) model around the predictions that needs interpretation. The purpose for LIME was to instil trustworthiness of a prediction. The paper also stated how explanations from LIME can be used for model selection by evaluating generalizability of a model. The study in (Ribeiro et al., 2016) used two product review sentiment analysis datasets consisting of books and DVDs to analyse interpretability outcome on text classification. In the experimental set up, the study has trained multiple algorithm for text classification including transparent models like linear regression and decision tree and non-transparent models like SVM and Random Forest. Then important features from the transparent models are used to evaluate performance of LIME by comparing the important features extracted from explanation generated for non-transparent models. In general, interpretations generated from local explainability approach suffer from human bias as the technique need a separate model building and human interpretation of new model outcome. LIME specifically considered unstable as it creates sparse linear model locally which would not be able to fit the observations efficiently if observations are not linear locally. (Villarroya, 2018) has adapted LIME for explaining DNN based NER model which is discussed in detail later in this section.

SHAP (SHapley Additive exPlanations) was introduced by (Lundberg and Lee, 2017) where the methodology unifying existing methods - LIME, DeepLIFT, LRP and classical Shapley values from game theory. Kernel SHAP was proposed as Linear LIME + Shapley values where SHAP could avoid the uncertainty of heuristically choosing loss function, weighting kernel and regularization term which was the case for LIME. Apart from Kernel SHAP, the paper also proposed Linear SHAP for linear models, Low-order SHAP where number of features are less, Deep SHAP (DeepLIFT + Shapley values) specific to DNNs. In DeepSHAP, it could avoid the need to heuristically choose appropriate rule ("*Linear*" vs "*Rescale*" vs "*RevealCancel*") for calculating contribution scores at nonlinear functions (Shrikumar et al., 2017). (Robinson and Yufeng, 2019) and (Li, 2019), in their blogs demonstrated the use of DeepSHAP on DNN based text classification to generate global explainability to build user trust on the model.

2.2 AI Explainability for NER

Literature review suggests that AI explainability in NLP have been mainly focused on explaining predictions for classification models. In recent past, ML explainability techniques were studied on NER models only in couple of cases. (Xie et al., 2018) demonstrated the use of LRP to interpret the Bi-LSTM-CRF model used in Named Entity Recognition. Primary purpose of the paper was to investigate different explainable deep learning methods for NER, implement an appropriate explainable method for the named entity recognizer based on Bi-LSTM-CRF, and explain the NER model to people having some background knowledge using the visualization. This paper used t-SNE visualization for word embedding and used LRP on Bi-LSTM-CRF to visualize how much each word contribute to the output and how contextual words are related. This study also tried to explain the wrong NER prediction using several examples. This paper also highlighted the need **to apply other explainability techniques on NER to validate the explanation generated using LRP.** The study also indicated the need of **further consideration on how to use AI explainability to analyse wrong NER predictions.** The result of the study lacks in 2 aspects (1) **Interpretability for users having language and domain understanding but no formal ML knowledge.** (2) Interpretability of CRF layer as use of LRP was focused on the Bi-LSTM layers as LRP is DNN specific feature relevance explainability approach (note that any gradient based explainability approach will suffer from this problem)

In another study, (Villarroya, 2018) demonstrated AI explainability on NER task using LIME. The primary goal from the study include generating explanation for NER task using LIME and evaluation of explanations for entities. This paper implements NER using a Bi-LSTM and generated explanation using two different techniques – LIME and LIME-NER which is a variant of LIME for explanation of sequence detection task. For evaluation, the study used a method named "*Area over the perturbation curve*" (AOPC) adapted for LIME-NER. The AOPC metric rely on a perturbation process similar to LIME, though they follow different strategy for sampling of the words to perturb. This paper indicates that use of **Kernel SHAP for explaining NER** would be a potential future work and also suggested use of the **LIME-NER technique on a different data set to understand generalization of this explanation techniques** for NER task.

This study will focus on the following shortcomings and future works suggested by previous studies performed on AI explainability for DNN based NER task.

- Kernel SHAP (Lundberg and Lee, 2017) for explaining NER as suggested in (Villarroya, 2018)
- NER explainability for users having sufficient language and domain understanding but no formal ML knowledge as suggested in (Xie et al., 2018)
- AI explainability to analyse wrong NER predictions as suggested in (Xie et al., 2018)

3 Research Questions

Literature review in previous section leads to following research question for the current study.

- *Is Kernel SHAP an effective method to explain Named Entity Recognition Model built using BI-LSTM-CRF architecture?* – as suggested in the future work of (Villarroya, 2018), this study would like to generate explanation for NER task using Kernel SHAP as suggested in (Lundberg and Lee, 2017) and compare the explanations generated using LIME-NER as suggested in same paper.
- *Are the explanations generated using Kernel SHAP for Named Entity Recognition Model easy to understand for users having language and domain understanding but no formal ML knowledge?* – The study by (Xie et al., 2018) is insufficient for generating explanation for NER model for users having language and domain understanding but no formal ML knowledge. This study would like to understand whether a methodology can be devised to address this gap.
- *Are the explanations generated using Kernel SHAP for Named Entity Recognition Model helpful in improving NER model performance?* - as suggested in the future work of by (Xie et al., 2018), this study would like to investigate how NER explainability generated using Kernel SHAP (Lundberg and Lee, 2017) can be used to identify insights from wrong NER prediction and use those insights to make improvements in NER model.

4 Aim and Objectives

The main aim for this study is to generate explanations for Named Entity Recognition (NER) predictions from state-of-the-art NER model (built using BI-LSTM-CRF architecture) using Kernel SHAP which is a variant of SHAP framework (Lundberg and Lee, 2017). The primary goal of this research is to generate explanations for NER predictions that are easy to understand for users having language and domain understanding but no formal ML knowledge

The research objectives are formulated based on the aim of this study, which are as follows.

- Generate explanations for NER prediction using Kernel SHAP (Linear LIME + Shapley values) (Lundberg and Lee, 2017)
- Evaluate explanation generated from Kernel SHAP (Lundberg and Lee, 2017) in following dimensions –
    1. General interpretability of explanations generated
    2. Insights from explanation for wrong NER predictions that can improve NER model prediction performance.
    3. Assessing quality of the model improvement insights by implementing them to revise NER model and then comparing model performance of initial NER model and revised NER models based on at least 2 least performing entities from initial NER model.

5 Significance of the Study

- The study will explore how AI explainability can be used to generate explanations for NER task using one of the most recent techniques in the AI explainability space. Kernel SHAP, as proposed by (Lundberg and Lee, 2017), have not been used to explain NER predictions so far. Therefore, this study intends to explore this new area in AI explainability domain.
- The study aims to provide comparison between explanations generated using LIME (Ribeiro et al., 2016) and Kernel SHAP (Lundberg and Lee, 2017) in following 2 dimensions - (1) Interpretability of explanations generated for users having no / limited ML understanding. The comparison of LIME and Kernel SHAP explanations in NER prediction will contribute to the knowledge base in AI explainability domain. (2) Insights to improve NER model prediction performance. Though this idea is suggested by (Xie et al., 2018), this study intends to perform this using LIME and Kernel SHAP and that will be a novel contribution for DNN based NER models.

6 Scope of the Study

The scope of the study will be limited to following items

6. NER model building on a subset CoNLL 2003 English named entity dataset (Language-Independent Named Entity Recognition (II), 2020) and validation of the NER model using sequence F1 score

7. Generating explanation using LIME-NER as implemented in (Villarroya, 2018) and visualization of explanations
8. Generating explanation using Kernel SHAP as proposed in (Lundberg and Lee, 2017) and visualization of explanations
9. Comparison of explanations generated from both the methods using the evaluation technique AOPC that is adapted by (Villarroya, 2018) for NER
10. Building a revised NER model using insights gathered from explanations on wrong predictions and compare performance of the revised model against the initial NER model.

The following items will not be in scope for the study.

1. Building NER model on complete CoNLL 2003 English named entity dataset (Language-Independent Named Entity Recognition (II), 2020) as the available infrastructure will not be suitable for the same. This is discussed in detail in section [3.2.2].
2. Generating explanation using other SHAP method - Deep SHAP (DeepLIFT + Shapley values), even this technique might be suitable due to the DNN architecture of the NER model.

7 Research Methodology

7.1 Introduction

Based on the goal of the current research, this can be categorised as **Applied Research** as the aim of the study is to assess methodology of AI explainability on NER models.

Based on objective, the current research can be categorised as **Descriptive Research** as the study will try to answer **how** Kernel SHAP can be used to generate explanation for NER models, **how** explanations generated from Kernel SHAP would be comprehendible to users and **how** the explanation generated for NER can help improving model performance.

Based on the type of information sought, this research can be classified as **Quantitative Research** as the study will be employing statistical methods like Kernel SHAP which combines LIME and classical Shapley values from game theory for generating explanations for NER model.

7.2 Dataset Description

This research will use the CoNLL 2003 English named entity dataset (Language-Independent Named Entity Recognition (II), 2020) which is the same dataset used by (Xie et al., 2018). This is a public dataset containing every sentence is tokenised in words. Each word present in new line along with space separated parts of speech (POS), chunk name (e.g. noun phrase, verb phrase, pre-positional phase etc.) and named entity tags. Each sentence in a single document is separated by an empty new line. The table below shows the size of sentences and words in training, test-a and test-b datasets. A direct downloadable copy of this data is available at following GitHub location: https://github.com/synalp/NER/tree/master/corpus/CoNLL-2003

Data Set	Number of Documents	Number of sentences	Number of words
Training	946	14987	203621
Test-a	216	3466	51362
Test-b	231	3864	46435

Table 44: CoNLL 2003 English named entity dataset details

The data set contain 4 different entities – LOC: location, PER: person, ORG: organisation and MISC: miscellaneous. The data set follows BIO tagging scheme where beginning word of an entity is tagged as 'B-<entity name>', e.g. 'B-PER'; any subsequent word that is part of the entity is tagged as 'I-<entity name>', e.g. 'I-PER'. All other words that is not part of the entity is tagged as 'O'.

7.3 Data pre-processing

- Identify training subsample – It will be required to select a subsample from the training set as it is unlikely that the complete training data can be used to train the NER model in available infrastructure. The plan is to create a subsample based on density of entities in the sentences, i.e. if a sentence does not have any entity of interest then that will be excluded in first iteration of subsample selection.
- Spelling related feature creation – Feature creation step in (Huang et al., 2015) mention several spelling related feature generations so that the features can be directly fed to the CRF layer bypassing the Bi-LSTM layers. Below is the subset of spelling related features from the paper that is planned to be used in this study.

- is first letter of the word in upper case
- are all letters of the word in upper case
- are all letters of the word in lower case
- is any letter in the middle of word in upper case
- does the word contain both letters and digits
- does the word contain punctuation
- word pattern – all upper-case letter replaced by 'A', all lower-case letter replaced by 'a', all digits replaced by '0', all punctuations replaced by '-'
- word pattern summarization - all consecutive upper-case letter replaced by 'A', all consecutive lower-case letter replaced by 'a', all consecutive digits replaced by '0', all consecutive punctuations replaced by '-'

- Context related feature creation - Feature creation step in (Huang et al., 2015) mentions use of POS and CHUNK as context related features. Same is planned to be used in this study as well.

7.4 Transformation

- **Word to word index transformation** - The NER model will be built using a BI-LSTM-CRF architecture where the words will be the inputs to the Input layer of the DNN. However, words cannot be directly fed to the input layer, therefore each word from the vocabulary of training set will be converted to a number. In order to reduce rare words and the words not present in GloVe embedding, plan is to retain only those words that are either available in GloVe embedding or occurs more than 3 times in the training subsample.
- **Variable to fixed sentence length -** Sentences in the training corpus does not have a fixed length. Therefore, it is planned to use a fixed length of words in the sequence. The strategy will be to calculate summary statistics for sentence length and to use the 3rd quartile value as the fixed sentence length for training. Any sentence having more words than the fixed length will be truncated till the fixed length and any sentence having a smaller number of words than the fixed length will be padded with a fixed dummy word towards the end till the sentence length reaches the fixed length.
- **Word Embedding initialisation with GloVe** (GloVe: Global Vectors for Word Representation, 2020) – Word index inputs will be fed to an embedding layer.

To achieve better performance, strategy will be to use pre-trained GloVe embedding to initialise the weights of embedding layer.

7.5 Models

Below flow diagram depicts planned sequence of activities in the modelling and explanation generation phase.

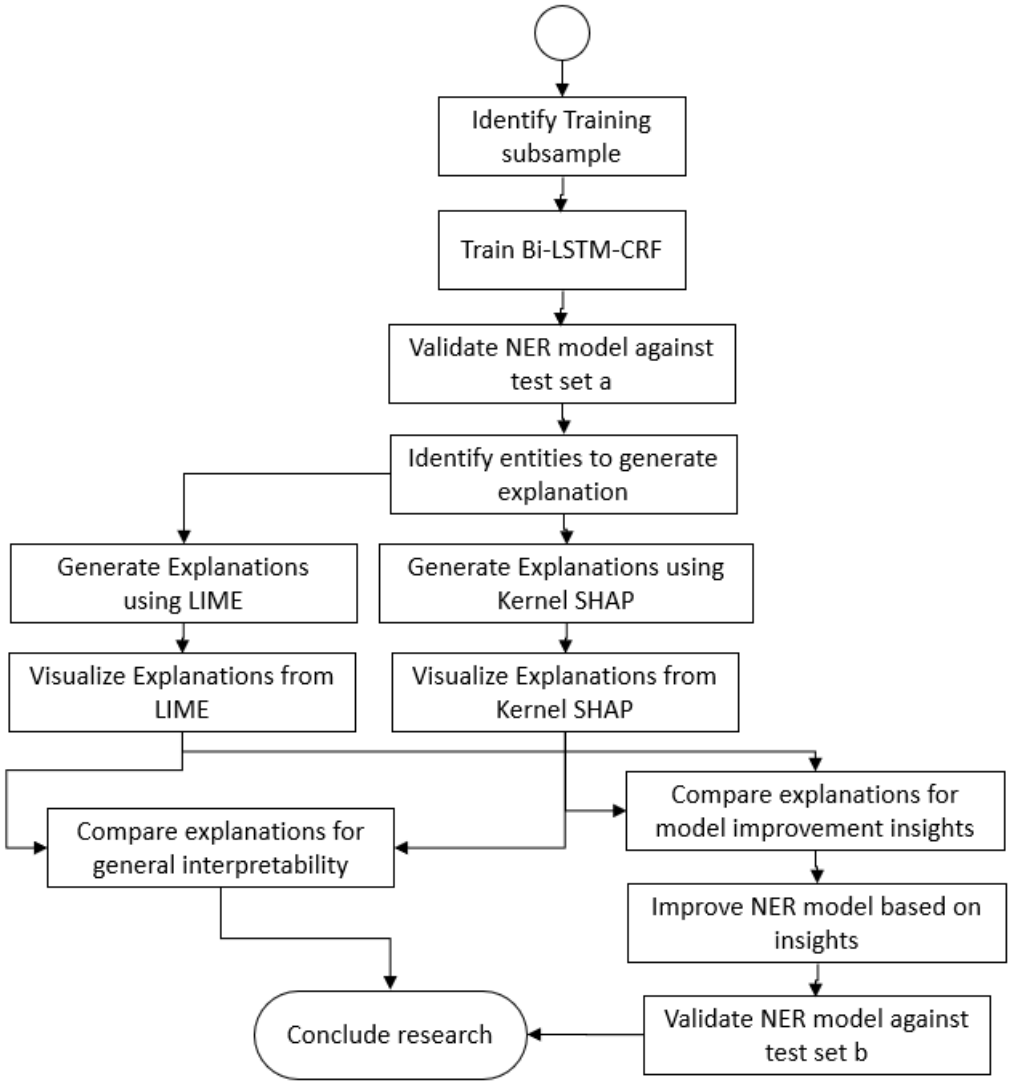

Figure 73: High-level research approach

- **NER model building** – shall build Bi-LSTM-CRF for NER tag prediction as proposed in (Huang et al., 2015).
- **Validate NER model** – The NER model will be validated against test set a – there are two test sets available with CoNLL-2003 (Language-Independent Named Entity

Recognition (II), 2020). The study shall use sequence F1 score as primary performance metrics for validation of NER tag prediction.

- **Identify entities to generate explanation** – 2 best performing entities and 2 worst preforming entities as per sequence F1 score of an entity.
- **Generate explanations as "Local Interpretability"** using following techniques
    - LIME (Ribeiro et al., 2016) – will use LIME-NER as suggested by (Villarroya, 2018)
    - Kernel SHAP (Lundberg and Lee, 2017) – will use PyPI shap library to generate the explanation on NER model.
- **Visualize Explanations** generated form above step – In this step it will be necessary to trace back to features related to actual words which have been abstracted by embedding and Bi-LSTM layers in the network.
    - LIME (Ribeiro et al., 2016)
        - Multiclass Variable importance plots – this will help visualising positive and negative impact of different features for each pair of possible classes.
    - Kernel SHAP (Lundberg and Lee, 2017)
        - Variable importance plot – This helps in visualising mean absolute value of the SHAP values for each feature globally. This plot helps in global interpretation of a model.
        - Partial dependence plot – This helps in visualising which features are most important for a model. It is possible to plot the SHAP values of every feature for every sample.
        - Force plot - This helps in visualising features contributing to push the model output from the base value (in case of NER base value would 'O' which represents that word in not part of an entity) to the model output.

7.6 Evaluation metrics

- The initial NER model that will be built in order to generate explanations will be evaluated on test set a using sequence F1 score. Accuracy will not be an appropriate performance metrics as the proportion of 'O' tags (words not a part of an entity) are much higher than the number of entities in a sentence. Sequence F1 (Hironsan,

2019) will be more appropriate than F1 score for individual tags as it is more likely to have sequence of words as part of a single entity.

- Explanations generated from NER model using LIME and SHAP will be Compared in following dimensions
    - General interpretability – this is aligned to primary goal of the study. The study will use "*Area over the perturbation curve*" (AOPC) adapted for LIME-NER by (Villarroya, 2018). AOPC was proposed by (Samek et al., 2017a) for quantitative comparison of heat maps generated from multiple explainability technique (Sensitivity Analysis, LRP and deconvolution network) on multiple image classification datasets. (Villarroya, 2018) adapted APOC for NER explainability.
        - As suggested by that study, for correct NER predictions, APOC will be calculated by perturbing in *most relevant first (MoRF)* method. NER prediction probability score for ground-truth class will be calculated per entity after removing most relevant word from sentences iteratively as per the LIME and Kernel SHAP scores separately. The word will be removed by setting the embedding values to 0. This will be performed on all observations and NER prediction probability score for ground-truth class will be averaged in each iteration of word removal.
        - For in-correct NER predictions, APOC will be calculated using *least relevant first (LeRF)* perturbation process which is intuitively reverse of MoFR. Here the only difference from earlier method will be that NER prediction probability score for ground-truth class will be calculated per entity after removing least (instead of most) relevant word from sentences iteratively as per the LIME and Kernel SHAP scores separately.

      It is expected that the initial steepness (that represents NER prediction probability score of ground-truth class) of the AOPC achieved by implementing above strategies will be higher for the better NER explanation technique (LIME vs Kernel SHAP).

      It is also expected that variation of NER prediction probability score of ground-truth class averaged over all observations (represented by

AOPC) will be higher for the better NER explanation technique (LIME vs Kernel SHAP).

The study also intends experiment on strategies that can improve upon the evaluation criteria so that in can capture subjectivity in explanation generated.

- Model improvement insights – Study intend to compare the insights from explanations generated from the 2 methods so that the NER performance on selected poor performing entities can be improved

- The plan for the study includes establishing whether model improvement insights are valid by improving the earlier NER model based on insights gathered form generated explanations on wrong predictions. Once the NER is model is updated based on the insights, old and new NER model will be validated on test set b by comparing entity recognition performance using sequence F1 score.

8 Requirements / resources

8.1 Hardware Requirements

In this study, experiments will be performed on a PC with following configurations:

- Intel Core i5-850U CPU, 1.6 GHz, 4 cores
- 8 GB DDR RAM
- Windows 10 64-bit OS
- NVIDIA GeForce MX-150 GPU with 2 GB dedicated memory

8.2 Software Requirements

Experiments will use GloVe embedding (with 300 dimensions) (GloVe: Global Vectors for Word Representation, 2020) for initialising embedding weights.

The Bi-LSTM-CRF model will be implemented using Keras and Python. Keras is a python library that provides APIs to build deep learning models using TensorFlow as backend.

The experiments will use text processing open source libraries form NLTK, GENSIM and Sci-kit learn. For generating explanations, the study will use lime (Ribeiro, 2016) and shap (Lundberg, 2017) libraries from PyPI.

9 Research Plan

Project Planner - XAI in NER

Select a period to highlight at right. A legend describing the charting follows.

Period Highlight: 1 | Plan Duration | Actual Start | % Complete | Actual (beyond plan) | % Complete (beyond

ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Literature search	1	4			75%
Milestone 1 - Literature review	1	6			50%
Get CoNLL-2003 data for NER	1	0.5			100%
Identify Training Subsample	1	0.5			0%
Design BI-LSTM-CRF	1	0.5			0%
Training BI-LSTM-CRF	2	0.5			0%
Validate NER model against test set a	2	0.5			0%
Identify entities and sample observations to generate explanation	3	0.5			0%
Use LIME to generate explanations	3	0.5			0%
Milestone 2 - Visualize explanations generated from LIME	4	0.5			0%
Use Kernel SHAP to generate explanations	4	0.5			0%
Visualize explanations generated from Kernel SHAP	5	1			0%
Defining Evaluation method for comparing explanations	6	1			0%
Comparing general interpretability of explanations form 2 techniques	7	1			0%
Milestone 3 - Finalization of Interim report	8	1			0%
Extract insights from Kernel SHAP explanation for NER model improvement	9	2			0%
Adjust Data / NER model design based on model improvement insights	11	1			0%
Milestone 4 - Retrain NER model	12	0.5			0%
Validate initial NER model against test set b	12	0.5			0%
Validate revised NER model against test set b	13	0.5			0%
Compare NER performance between both version of NER models	13	0.5			0%
Milestone 5 - Finalizing Final report	14	2			0%
Final Delivery - Slides and video preparation	16	1			0%

PERIODS - Weeks

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33

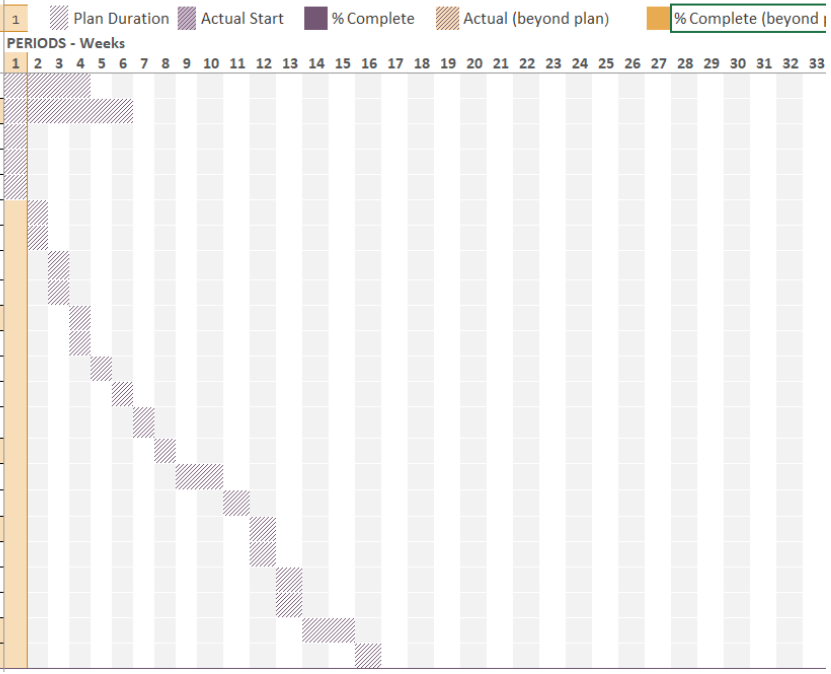

Figure 74: Research timeline and milestones