Radboud University Nijmegen

Faculty of Science

Evaluating The Use Of Entity Triggers As Explanations In Named Entity Recognition

Thesis BSc Computing Science

Author:
Yori (Y.Y.) Jonkman

First supervisor/assessor:
prof. Martha (M.A.) Larson

Second assessor:
dr. Iris (I.H.E.) Hendrickx

Second supervisor:
Jane (J.A.) dela Cruz

March 26, 2024

Abstract

In this thesis, we explore a new use of entity triggers in named entity recognition. Entity triggers are groups of words used by a NER model to efficiently improve performance by informing context. Under normal use, entity triggers are only used internally by the model and are not seen by a user of the model. We extract information about the entity triggers in order to inform the user about the process going on within the model during prediction. This makes the model more transparent and easier to explain. Our research focuses on applying these explanations in disaster risk management during the identification of locations mentioned in posts on the social media website Twitter. The information which is extracted from the model includes a trigger key and a distance. The trigger key is a textual representation of the learned entity trigger which is deemed most similar to an entire prediction sentence by the model. We find trigger keys not to be intuitive in their relation to prediction sentences. Chosen trigger keys often have no lexical overlap and little semantic relevance. This is the case even when among the unchosen trigger keys exists a key with excellent similarity to the prediction sentence in the eyes of a human user. The distance is defined as a numerical value describing the semantic similarity of a prediction sentence to the most similar entity trigger from the training data. We find that the distance can be used as a heuristic to determine the semantic familiarity of a prediction sentence given the data the model was trained on. This familiarity is particularly computed from the first and last tokens of the sentence. We find that entity triggers offer little additional insight on the labelling decisions made by the model. Trigger keys give results with no relevance to the prediction sentence. The importance placed on the first and last tokens by the distance correspond poorly to how humans would pick out named entities from a sentence. Our findings suggest the model picks out named entities using information that is different from information a human would use. Although entity triggers improve performance of the NER model, these unintuitive explanations may indicate faulty reasoning underneath the hood. Future work should investigate explanations using other methods than trigger keys and distances. Additionally, the internal reasoning using entity triggers should be closely examined to determine flawed reasoning.

Table of Contents

1 Introduction

1.1 Motivation

The field of disaster risk management focuses on minimizing damage done by natural disasters. In this pursuit, artificial intelligence is able to offer assistance through techniques such as named entity recognition. Named entity recognition is used to identify specific classifications of phrases in unstructured text. One such application is the detection of locations mentioned in social media posts. This can be utilized in disaster risk management to identify locations with vulnerabilities and requests for aid.

As this utilization of artificial intelligence plays a weighty role in vulnerable human lives, it is of importance that the decisions made in response to the results of the artificial intelligence model are well-informed and held to accountability. To help do so, the model should be transparent in its decisions. By justifying each of the model's results with an explanation, a user can more easily interpret the decision-making process followed by the model.

In this paper, we explore the possibility of generating additional explanations for a specific named entity recognition model. We focus on location detection for disaster events in posts on the social media site Twitter. The model we build off of is TriggerNER, developed by B. Lin et al. in 2020 [1]. This model is designed to utilize a technique proposed by B. Lin et al. which they dubbed "entity triggers." Entity triggers are a group of words which help to explain why humans consider a specific phrase in the same sentence to belong to a particular class. In the case of this paper, that class would be a location. Examples of entity triggers for a location may be the phrase "welcome to" or "living in" preceding the location. The TriggerNER model uses these entity triggers to efficiently improve its contextual awareness and performance.

While TriggerNER utilizes entity triggers to help improve the model, none of the calculations or decision-making process are communicated to the user. The model only informs the user of the final classification of each word determined in sentences provided by the user. We explore the possibility of using entity triggers to aid the user's understanding of the underlying process as well.

The explanations we extract from entity triggers are a trigger key k_d and a distance d for each prediction made by the NER model. A trigger key is a textual representation of the entity trigger deemed by the model to be the most similar to the prediction sentence. A distance is a numerical representation of how similar the chosen entity trigger and the prediction sentence are to the model. Thus, the output of our extended TriggerNER model when asked to predict locations in the sentence "Rohingya Refugees extend helping hand to Kerala Flood Victims" may look like this:

$$\begin{array}{c} \text{Rohingya Refugees extend helping hand to } \underline{\text{Kerala}} \text{ Flood Victims} \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{LOC} \end{array}$$

$$k_t = \text{donating}$$
$$d = 0.7780$$

Where k_t is the trigger key and d is the distance.

1.2 Research Questions

The main question addressed in this thesis is formulated as follows:

How can entity triggers add value as explanations for named entity recognition in disaster risk management?

This question is answered with the aid of several sub-questions:

1. What is the difference in performance between algorithms trained on Tweet data annotated with entity triggers and trained without entity triggers?
2. What explanations are offered to justify tagging an entire Tweet as containing a location entity?
3. Are the offered explanations intuitive?
4. What explanations are offered to justify tagging an entire Tweet as **not** containing a location entity?

Answering the first sub-question will aid in the understanding of the value of entity triggers in the training process. Identifying the cause for differences in performance may further help to find the value of entity triggers as explanations.

The second sub-question's answer will give the user insight to the model's decision-making process. By examining the explanations generated by the algorithm, the model's choices should become easier to interpret. Additionally, discovering which explanations the model is able to offer tells us what information the model uses in its decision-making process.

Answering the third sub-question will aid in improving the explainability. Intuitive explanations offer better understanding of the model's decision-making process. These explanations are then more valuable to users as well. If some explanations are intuitive while others are not, we may be able to identify logic errors in the model, which can then be tweaked.

The fourth sub-question identifies why a false negative may occur, in addition to true negatives. Together with the answers from the preceding sub-questions, the AI model may be fine-tuned to avoid the circumstances in which false negatives occur and promote those where locations are identified, such that location entities are identified with greater accuracy. As our research focuses on the identification of locations in need to support from disaster risk management services, we wish to minimize false negatives such that no persons in need are missed by the algorithm.

In order to answer these questions, we explore the TriggerNER model and create our own dataset to train it on in section 3. We run several experiments on a trained TriggerNER model to evaluate our possibilities in section 4. The results to these experiments will be analyzed in section 5 in order to determine our answers.

2 Literature Review

2.1 Disaster Risk Management

Natural disasters are responsible for thousands of casualties and billions of economic losses [2]. The field of disaster risk management (DRM) strives to minimize such damages in four phases: mitigation, preparation, response and recovery [3]. Of these four phases, response focuses on minimizing damage while a disaster is ongoing. A rapid response allows for the swift identification and rescue of affected people, which can save lives. Furthermore, first-aid and humanitarian assistance can be provided more efficiently.

In recent years, artificial intelligence (AI) techniques have been utilized to improve response rate [3]. One such application involves supervised machine learning (ML) models which have been trained to detect likely areas of impact. As there has been a trend in information critical to the disaster relief effort being posted to social media websites, some models use social media sites such as Twitter in order to determine affected areas [4, 5]. One technique utilizes geographical data embedded in the metadata

of the online post to determine areas [6]. Locations may also be mentioned in the content of the post itself. To efficiently extract mentioned locations from the content, AI uses Information Extraction (IE) methods such as named entity recognition (NER).

2.2 Named Entity Recognition

NER has been defined as one of the fundamental sub-tasks of IE [7]. NER aims to identify units of certain classes of information from unstructured text, such that they can be extracted and utilized in structured applications. In the previously mentioned use case, the class of information NER seeks to identify is locations. For instance, in the Twitter post "Bangladesh donates US $500000 for Sri Lanka flood victims #news" the AI should recognize the words "Bangladesh" and "Sri Lanka" as distinct locations with the use of NER. These relevant phrases are referred to as named entities.

NER models are first trained on a large corpus of sentences, which has the desired classes of information within it labelled by hand [7]. After encountering them in the training data, the model learns to recognize relevant words and other sentence components such as punctuation. A word, punctuation character, or another sentence component is referred to as a token. The model can then efficiently pick out these tokens from unlabelled sentences. This is referred to as supervised learning.

In some models, tokens which were not seen in the training data but still belong to the same classes have their class inferred from contextual clues encountered in both the training data and the unlabelled sentence. This is referred to as semi-supervised (or weakly supervised) learning. However, false positive matches also become more common.

Beside the training data, NER models can utilize a validation dataset during training. Once the model is trained to recognize named entities, it may evaluate how well it performs when trying to predict the named entities in its validation dataset. In order to improve the model's performance this is done several times, tweaking the model slightly each time to try and improve accuracy. Datasets which are used to evaluate the performance of the NER model after training is done are referred to as test datasets. Test data is not used to tweak the model, merely to evaluate it.

It should be noted that NER annotation among datasets can follow different, conflicting annotation guidelines. Additionally, annotation may not always follow the intended guidelines. The behavior of a NER model is affected by the annotation patterns found in its training data and verification data. The evaluated performance is affected by the patterns found in the verification data and test data. Incorrectly tagged named entities in these datasets can produce false positives and false negatives. The annotation patterns among the datasets used in our research will affect our results as well. We acknowledge that the models evaluated in this paper are trained on crowdsourced data intended to follow guidelines established by Suwaileh et al. [8].

Research into various methods to improve the accuracy of NER algorithms has been done over the years [7, 9]. The work in this paper builds off of the methods developed by B. Lin et al. for TriggerNER [1]. TriggerNER improves the semi-supervised technique by utilizing "entity triggers."

2.3 Entity Triggers

Entity triggers are defined by B. Lin et al. as groups of words which explain why a named entity belongs to a specific class [1]. These entity triggers should be sufficient for a human user to infer which class is being referred to without seeing the named entity.

By utilizing entity triggers in conjunction with standard named entities, the performance of NER models is improved. Entity triggers give a model more context to

work off of, enhancing its generalizability. Named entities which were not present in the model's training data may be classified more easily with the aid of the semantically similar sentences surrounding them. In addition, entity triggers are more cost-effective than training with just named entities. In the paper proposing entity triggers [1], experiments showed that model performance using only 20% of trigger-enhanced sentences was comparable to model performance using 50-70% of sentences with just named entities, while annotating entity triggers in addition to named entities for the training data was estimated take only 1.5-2 times as long as only annotating named entities. Entity triggers thus enable the training of a more generalizable and cost-effective NER model.

In this paper, we will represent named entities by underlining them once and placing their class beneath this line. We represent associated entity triggers by underlining them twice. One example of a sentence containing both a named entity and associated entity triggers is:

We had a fantastic lunch at Rumble Fish yesterday, where the food is my favorite.
RES

Here, the words "had [...] lunch at" and "where the food" help to infer the class of Rumble Fish, in this case a restaurant (RES). This paper will focus on locations (LOC).

2.4 Explainable Artificial Intelligence

AI has its limitations. Training data is limited and may introduce bias, privacy and security risks [10, 11]. The decision-making process of models is often not transparent and explanations are difficult to interpret or incomplete [12]. Given the weighty role DRM plays in human lives, it is of importance to be mindful of these limitations and minimize risks when integrating AI into DRM.

Improving the explainability of AI helps to circumvent and minimize these risks. By researching and adhering to explainable AI (XAI) design patterns, the decision-making process of models becomes more transparent. With interpretable and complete explanations for AI decisions, it becomes easier to identify risks for bias and security. In addition, it is easier to identify the limitations of AI and mitigate inflated expectations [11].

As entity triggers utilize human explanations in the NER training process itself, we hypothesize that entity triggers can be used to enhance the explainability of NER models. We aim to generate explanations for the AI's classification decisions by extracting the semantically-related trigger phrases identified by the AI in its predictions.

3 Implementation

3.1 Style Guide

We define guidelines to assist in the decision whether to tag a token as a entity trigger. The main purpose of entity triggers is to explain why their related named entity belongs to the class it does. In the case of this paper, entity triggers explain why a named entity is a location specifically. We aim to utilize entity triggers as explanation on their own without further justification. Thus, when entity triggers are tagged for dataset creation, a person examining the dataset should be able to understand why a entity trigger was chosen without confusion or need for elaboration. This should allow us to take a step towards being able to use entity triggers to improve the explainability of the system trained on them.

No guidelines for annotating entity triggers exist yet, so we develop our own. These guidelines were chosen based on the intuition of the author and common patterns noticed

by the author in existing trigger-annotated datasets [1]. More research should be be done in determining which patterns often make for good trigger explanations.

Entity triggers must be associated with a single named entity. These guidelines assume that the sentence being annotated only has one named entity in it. If a sentence has multiple named entities, the sentence should be duplicated for each named entity. Each duplication should have one unique named entity each.

It should be noted that it is possible there are no tokens suitable for explanation in a sentence. In this case, we simply do not tag any entity triggers. Additionally, should we come across an entity in our dataset which was tagged as a location entity despite not being a location, we do not tag any entity triggers as well. We write down the index of these falsely-tagged entities so that we may choose to cull these entries at a later time.

1. Tag phrases which define the named entity. If such phrases are known to be a location, it is easily inferred that the named entity is a location. Examples:
   - Local currency depreciated in countries such as Brazil, Chile and South Africa.
     LOC
   - New York is a bustling city. It is nicknamed "The City That Never Sleeps."
     LOC
2. Tag phrases which describe features of the location. If a named entity is said to contain municipalities or mountains, it is likely a location. Examples:
   - Office locations were reduced in the US and Mexico.
     LOC
   - We eventually arrived in the indigenous community of Mechahuasca.
     LOC
3. Tag phrases which the location describes as an attributive noun. In other words, if the location acts like an adjective. Examples:
   - A Kansas woman bought the entire stock to donate to Nebraska flood victims.
     LOC
   - A Kansas woman bought the entire stock to donate to Nebraska flood victims.
     LOC
4. Tag verb phrases which describe events that imply a location. Verbs tend to refer to specific classes of entities: the object following a phrase such as "travelled to" is likely describing a location. Examples:
   - Celebrities quickly evacuate California as wildfires rage.
     LOC
   - We eventually arrived in the indigenous community of Mechahuasca.
     LOC
5. Tag prepositions referring to the location. Prepositions often denote a direction or spatial relationship. This helps to infer locations, particularly together with verbs from guideline 4. Examples:
   - A mountain biking race from Luxembourg to the Netherlands collects donations.
     LOC
   - Begging 4 help in Canoa, pacific coast. There is no way to get there by road.
     LOC

6. Tag other locations in the same clause as the named entity. For instance, if a named entity in among a list of locations, it is likely a location itself as well. Examples:
   - Local currency depreciated in countries such as Brazil, Chile and South Africa.
     LOC
   - Begging 4 help in Canoa, pacific coast. There is no way to get there by road.
     LOC
7. Tag pronouns referring to the named entity. All types of pronouns may aid in inferring the class of an entity, including but not limited to personal pronouns like "it," demonstrative pronouns like "that" and relative pronouns like "which." These pronouns may help distinguish inanimate locations from animate entities with similar names, such as a person named "Virginia" being referred to by "she" or "who" in the sentence. Examples:
   - New York is a bustling city. It is nicknamed "The City That Never Sleeps."
     LOC
   - Begging 4 help in Canoa, pacific coast. There is no way to get there by road.
     LOC
8. Tag adjectives describing details of the location. Certain adjectives are often used with locations specifically. Examples:
   - We went on a holiday to picturesque Florence for our honeymoon.
     LOC
   - New York is a bustling city. It is nicknamed "The City That Never Sleeps."
     LOC
9. Tag punctuation tokens only if they are part of words which help to infer the class of the named entity, such as periods or hyphens in words. Enclosing symbols such as parentheses or quotation marks are not considered part of the words they enclose. Examples:
   - 70,000 tarps to the U.S. Virgin Islands and Puerto Rico since Hurricane Maria.
     LOC
   - Storm Irma moving from Saint-Martin to Florida.
     LOC
10. Do not tag adjectives describing things other than the location as in guideline 8. Example:
    - Celebrities ~~quickly~~ evacuate California as wildfires rage.
      LOC
11. Do not tag punctuation unless it fits guideline 9. Ignore other punctuation such as parentheses, hashtags, commas, periods.
12. Do not tag parts of hashtag names. As hashtag names include no spaces, only tag these if the entire hashtag name is the entity trigger. Do not tag the hashtag symbol itself. Example:
    - Call if you wanna donate something #Amatrice #Accumoli ~~#PrayForItaly~~
      LOC

3.2 Dataset Creation

We construct a dataset of Twitter posts (Tweets) annotated with entity triggers and location named entities. This dataset focuses on location mentions in Tweets surrounding disasters in order to enhance applicability in DRM.

3.2.1 IDRISI-RE

We utilize an existing dataset on location mentions in disaster Tweets developed by Suwaileh et al. for IDRISI-RE [8]. The IDRISI dataset includes Tweets on the topic of 41 major disaster events ranging from 2016 to 2020, categorized by event. It covers 77,196 Tweets, of which 20,514 had their annotation crowdsourced through an online crowdsourcing platform and 56,682 had their annotations automatically labelled by AI trained on the crowdsourced Tweets. Each crowdsourced Tweet was labelled by three to eight persons and the final labels were selected from spans which were labelled by at least two persons. Annotators also had to maintain an accuracy above 70%. Tweets cover a variety of countries and disaster types (hurricanes, earthquakes, floods, wildfires and cyclones). Named entities have their location type specified (e.g. country, city, neighborhood). The dataset includes Tweets in English and Arabic.

3.2.2 Extending to a Trigger Dataset

We extend the IDRISI-RE dataset to include entity triggers. Each entity trigger is manually annotated by the author. Due to time constraints, we only annotate the English training data of crowdsourced Tweets from flood events, covering Sri Lanka (2017), Kerala (2018), Maryland (2018), and the midwestern U.S. (2019). This gives us a dataset of 2,289 unique Tweets containing 3,129 named entities. In order to distinguish which named entity groups of entity triggers belong to, we split each Tweet with multiple named entities into multiple duplicate Tweets with only one unique named entity per entry. This results in a dataset of 3,504 entries.

To tag the entity triggers in our dataset, we use the annotation software Doccano v1.8.4 [13]. We convert the IDRISI-RE dataset to a format which can be imported into Doccano with a simple Python program. We opt to keep each named entity as the ambiguous "location" class rather than specific classes such as "country," "city," et cetera. The Tweet contents are not modified. We manually go over each entry and tag spans of text denoting entity triggers in accordance to the style guide defined in section 3.1. Our final result contains 5,119 entity triggers. Each entity is given as a span of text, denoted with a starting and ending index as well as a string of text giving the class (location or trigger).

Before annotating the dataset, an entry may look like this:

Spreading smile in #Kerala (LOC). Distributing relief material to the people of #Kerala (LOC).

As previously mentioned, entries with multiple named entities are duplicated with only one unique named entity preserved respectively. Triggers corresponding to the one named entity are annotated. After annotation, an entry may look like this:

Spreading smile in (trigger) #Kerala (LOC). Distributing relief material to the people of #Kerala.

Spreading smile in #Kerala. Distributing relief material to the people of (trigger) #Kerala (LOC).

The resulting dataset and all used code is made publicly available in section 6.2.

3.3 Adjusting TriggerNER

We extend the TriggerNER model to output information aiding us in determining the explainability of NER decisions. The adjusted model will be used in section 4 to test the dataset created in section 3.2.

3.3.1 TriggerNER

TriggerNER is a NER model developed by B. Lin et al. to improve the accuracy of NER predictions in a cost-effective manner by utilizing triggers [1]. Before training, the text data must have all entity triggers manually annotated for each respective named entity. In the paper proposing TriggerNER by B. Lin et al., the authors crowdsourced this task. They masked each named entity with their respective class and gave annotators the instructions to mark "a group of words that would be helpful in typing and/or detecting the occurrence of a particular entity in the sentence," taking the intersection of three annotators' results to consolidate the final dataset. As detailed in section 3.2.2, we do not mask entities and have a single annotator for our entire dataset.

During training, each entity from the dataset is paired with its sentence and a group of its entity triggers. All tokens found across the sentence are encoded as numerical vectors. This encoding utilizes a convolutional neural network (CNN) to encode information on the characters of the token, which helps to pick out morphological information such as the prefix or suffix of a word [14]. By applying a bidirectional LSTM (BLSTM) neural network on these vectors, information on preceding and upcoming tokens are encoded into a new intermediate set of token vector. This information on surrounding tokens aids in considering context. Subsequently, these intermediate vectors are fed into a layer of conditional random fields (CRF) which helps inform which labels are suitable for each token. The resulting set of vectors are referred to as hidden states and are joined into a matrix. This hidden state matrix is used to compute a weighted attention vector using a self-attentive embedding developed by Z. Lin et al. [15]:

$$\vec{a_T} = \text{softmax}\left(W_2 \text{tanh}\left(W_1 H^T\right)\right)$$

Where $\vec{a_T}$ is the attention vector, H is the hidden state matrix and W_1 and W_2 are weight matrices which are dynamically updated during training to find the appropriate weighting. The softmax function ensures the total weights sum up to 1. By multiplying the weighted attention vector with the hidden state matrix, the final vector g_{Ts} representing the weighted sum of the token vectors in the entire sentence is obtained.

The same process is repeated for just the tokens found in trigger groupings instead of all tokens across the sentence. One small change is made to these trigger groupings during encoding: all numerical characters are generalized by replacing them with a generic "digit" character. This generic digit character is "0". After this process is complete, the final trigger vector g_{Tt} is obtained and fed into a multi-class classifier to predict the classification of the associated named entity. Each vector g_{Tt} is associated with a string of words to distinguish the vectors, known as the trigger key k_t. This string consists of the tokens in the original entity trigger. In the instance multiple entity triggers end up mapping to the same g_{Tt} values, a randomly-picked vector is preserved with its key while duplicate vectors are removed along with their keys.

After training is complete, these final vector spaces can be utilized in predicting the classes of token groupings of unlabelled sentences. The unlabelled sentence is encoded in the same way as during training in order to obtain g_{Ps}. TriggerNER then computes the distance d between g_{Ps} and every learned trigger vector g_{Tt} to determine the closest trigger vector, g_{Pt}. Then g_{Ps} and g_{Pt} are used to compute a weighted attention vector using the formula:

$$\vec{a_P} = \text{softmax}\left(\text{tanh}\left(W_1 g_{Ps}^T + W_2 g_{Pt}^T\right)^T\right)$$

g_{Ps} is then multiplied with $\vec{a_P}$ to obtain g'_{Ps}. This vector g'_{Ps} can be used for conventional NER prediction and is more likely to produce a correct prediction due to its weighting with the aid of entity triggers.

3.3.2 Extending to Provide Explanations

We extend TriggerNER to output additional information on how the model makes the decision to tag certain phrases as location entities. We build off of the TriggerNER model training pipeline provided by Lee et al. for LEAN-LIFE, an annotation framework utilizing entity triggers [16].

We modify the prediction process of TriggerNER such that two additional variables are produced for each sentence to be predicted. These variables are k_t and d, respectively g_{Pt}'s trigger key and g_{Pt}'s distance to g_{Ps}. We explicitly call training sequence data to the CPU to circumvent a programming bug related to a change in the software underlying TriggerNER since the release of TriggerNER. We also patch a bug in the LEAN-LIFE pipeline which interprets the first character of a sentence as its only token, rather than each word and punctuation symbol. Finally, we patch a bug in TriggerNER which causes trigger keys to index words from the global word list rather than from the triggers' respective source sentences, which had turned trigger keys into unrelated strings of tokens.

We extract the distance d with the intent to use it as a hint to the confidence of the trigger prediction, where a smaller distance would indicate greater confidence. The trigger key k_t gives a human-readable interpretation of which trigger vector is most similar to the entire prediction sentence and consequently used by the model to aid the prediction process. Note that d and k_t relate to the entire sentence and not specific tokens or predicted named entities. We determine whether these values are useful explanations in section 5.

After training, the adjusted TriggerNER model produces output akin to the following:

$$\underset{\text{LOC}}{\text{Rohingya Refugees extend helping hand to Kerala Flood Victims}}$$

$$k_t = \text{donating}$$
$$d = 0.7780$$

The extended model and all used code is made publicly available in section 6.2.

4 Experiments

We perform several tests in order to evaluate the dataset and NER model created in section 3. In these tests, we review whether a TriggerNER model trained with entity triggers outperforms a comparable NER model trained without entity triggers, as claimed in the TriggerNER proposal paper. In the process, we evaluate the utility of our trigger-annotated IDRISI-RE floods dataset created in section 3.2.2. Additionally, the usefulness of the explanations provided by our extended TriggerNER model created in section 3.3.2 is determined.

4.1 Experiment Datasets

We train two NER models on our trigger-annotated floods dataset. The dataset in its raw format describes entities as spans of text with starting and ending indices. The NER models take tokenized strings of text as input, so the dataset is reformatted with

a simple Python script and tokenized using the NLTK word tokenizer [17]. The utilized Python script and tokenized dataset are made publicly available in section 6.2.

The NLTK word tokenizer splits tokens on whitespaces and punctuation symbols. Punctuation symbols are preserved, whereas whitespaces are not. Certain punctuation symbols including the period ('.') and hyphen ('-') are not split into separate tokens if they are found before the end of a sentence. This is done in order to preserve abbreviated and hyphenated phrases such as "U.S." and "flood-hit" as distinct tokens. As a resulting downside, a token at the end of a punctuated sentence which was not space-separated from the first token of its following sentence will be combined with its following token. For example, in the following sentence fragment, the two words and period "Palghat.All" are seen as one single token.

train	from	Ratlam	to	Palghat.All	15	tanks	are	filled	,
		LOC							
O	O	LOC	O	O	O	O	O	O	O

We make the decision that only tokens which wholly fit within the indexed span of text will be tagged as entity, whether it be a named entity or entity trigger. As a result, some entities may be lost during tokenization of sentences.

As entity triggers must be associated with a named entity, tokenized data which contain entity triggers but no named entity due to such entity loss during tokenization is culled from the training data. We find a total of 20 tokenized Tweets like this, bringing the training data down from 3,504 entries to 3,484 entries.

Among these 3,484 entries, there are 529 unique named entities and 1,909 unique entity triggers. As for tokens, there are 591 unique tokens among named entities and 1,409 unique tokens among entity triggers. We give the count of the most frequent phrases and tokens in the training data in tables 2 and 3.

We acquire additional data to use as validation data during training and test data during evaluation. The IDRISI-RE dataset includes validation data with annotated named entities for each event. We combine the validation datasets for each of the four flood disaster events present in the training data and split this combined data in half to produce our validation dataset and a test dataset. We refer to the validation dataset as the "dev" data.

To test the generalizability of our model, we also test on two recent events of different disaster types. We create a cyclone test dataset by combining the training and validation data of the Cyclone Idai (2019) event as well as a hurricane test dataset by combining the training and validation data of the Hurricane Dorian (2019) event.

The statistics of each dataset we use during training, validation and testing is displayed in table 1.

1a. Floods training data				
Disaster Event	Entries	Unique Tweets	Named Entities	Entity Triggers
Sri Lanka (2017)	429	321	365	366
Midwestern U.S. (2019)	1,166	752	1,104	2,047
Kerala (2018)	1,395	907	1,164	1,761
Maryland (2018)	494	301	473	887
Total	3,848	2,281	3,106	5,061

1b. Floods dev data		
Disaster Event	Entries / Unique Tweets	Named Entities
Sri Lanka (2017)	45	55
Midwestern U.S. (2019)	107	164
Kerala (2018)	9	7
Maryland (2018)	—	—
Total	161	226

1c. Floods test data		
Disaster Event	Entries / Unique Tweets	Named Entities
Sri Lanka (2017)	—	—
Midwestern U.S. (2019)	—	—
Kerala (2018)	120	141
Maryland (2018)	42	95
Total	162	236

1d. Cyclone test data		
Disaster Event	Entries / Unique Tweets	Named Entities
Cyclone Idai (2019)	1,038	1,348

1e. Hurricane test data		
Disaster Event	Entries / Unique Tweets	Named Entities
Hurricane Dorian (2019)	1,038	1,011

Table 1: Entity statistics of the datasets used during evaluation.

2a. Named entities	
Phrase	Count
Nebraska	708
Kerala	549
Maryland	258
SriLanka	108
Ellicott City	99
Iowa	76
Sri Lanka	76
India	53
kerala	42
srilanka	32
Omaha	22
Baltimore	17
Ernakulam	17
Missouri	17
Chengannur	16
Kansas	16
Kodagu	13
Alappuzha	12
South Dakota	12
Karnataka	11

2b. Named tokens	
Token	Count
Nebraska	710
Kerala	549
Maryland	261
City	112
SriLanka	109
Ellicott	107
Sri	84
Iowa	78
Lanka	77
India	53
kerala	42
srilanka	32
Omaha	23
Missouri	20
Baltimore	18
Dakota	18
Ernakulam	18
Chengannur	16
Kansas	16
County	15

Table 2: Top 20 most frequent named entities and tokens in the floods training dataset used during our experiments.

3a. Entity triggers	
Phrase	Count
in	303
Nebraska	107
Iowa	100
floods	89
Maryland	72
flooding	67
flood victims	59
flood	57
flooding in	57
to	54
state	47
from	46
flood relief	39
Flood relief	34
floods in	31
Missouri	29
people of	29
homes	28
South Dakota	28
roads	27

3b. Trigger tokens	
Token	Count
in	964
flood	277
flooding	222
of	204
floods	179
to	156
Nebraska	155
Iowa	124
relief	121
from	119
victims	93
City	89
Flood	89
Maryland	88
flash	80
state	74
Ellicott	71
across	57
Relief	57
areas	55

Table 3: Top 20 most frequent entity triggers and tokens in the floods training dataset used during our experiments.

4.2 Experiment Models

We use our tokenized training dataset to train two models and compare their performances. One model is the TriggerNER model we extended to provide additional explanations in section 3.3.2. We describe how this model works in section 3.3.

The second model is trained on the same training data without entity triggers. We use this model to compare the performance of our trigger-enhanced model to a standard NER model trained without entity triggers. This allows us to test the hypothesis stating that a trigger-enhanced NER model performs better than a standard NER model. This can also show us whether the style guide we composed in section 3.1 produces a trigger-enhanced training dataset which improves performance. We will refer to the NER model trained without entity triggers as the "Standard" model.

The Standard model's operation closely resembles the operation of the TriggerNER model in order to effectively test the impact of entity triggers on the models' performances. As described for TriggerNER in section 3.3.1, the Standard model also encodes tokens into vector representations using CNN, BLSTM, CRFs and attention weighing. The difference in the implementation of the Standard model is that no entity triggers are encoded nor play a role in weighing the resulting model.

The implementation of both models is built off of the model pipeline provided by LEAN-LIFE [16]. Experiments are run in the web-based interactive Python development environment of Jupyter notebook. The trained models and notebook are made publicly available in section 6.2.

4.3 Evaluation Methods

Once both the extended TriggerNER model and Standard model have been trained, we evaluate their respective performances on the same datasets. In order to quantify their performance, we compute the F1 score. This score measures the true positives, false positives and false negatives produced by the models. The formulae used to compute these scores are:

$$\text{Precision} = \frac{N_{\text{True positives}}}{N_{\text{True positives}} + N_{\text{False positives}}}$$

$$\text{Recall} = \frac{N_{\text{True positives}}}{N_{\text{True positives}} + N_{\text{False negatives}}}$$

$$\text{F1 score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The resulting score is a value between 0% and 100%, where a greater percentage indicates a greater accuracy of the model's prediction.

We also evaluate the usefulness of the explanations extracted by our extended TriggerNER model. The explanations provided for each predicted sentence consist of the trigger key k_t and distance d. The trigger key and distance are only computed for the entire sentence and do not relate to specific tokens or predicted named entities.

The trigger key k_t gives the string of tokens determined to be the most similar to the prediction sentence. We analyze whether this information is useful and interpretable to the user. We examine how often particular trigger keys show up in our predictions in order to determine whether explanations are unique. Additionally, we evaluate whether particular trigger keys are associated with greater accuracy than other trigger keys.

The distance d is the smallest distance found between all trained entity trigger vectors and the prediction sentence's vector. We gather the distance ranges, means and medians found for each test dataset. Each interval of distance we find is evaluated for its performance to determine what level of confidence a range of distances might offer.

We also evaluate whether distances may provide a measure of familiarity and whether particular trigger keys are closely associated with particular distances. In order to test different degrees of familiarity, we evaluate the performance of the TriggerNER model on a variety of additional test datasets: one built from all training data featuring entity triggers (training test), one built from unique occurrences of each entity trigger (entity trigger test), one built from all final trigger keys saved by the trained model (trigger key test) and one built from randomly generated alphanumerical gibberish (gibberish test). Note that the trigger key dataset is very similar to the entity trigger dataset. The trigger key dataset is a subset of the entity trigger dataset with numerical symbols converted to a generic digit represented by zero. The sizes of the datasets used to test for familiarity are given in table 4.

Distance familiarity test datasets	
Dataset	Entries
Training test	1,742
Entity trigger test	1,909
Trigger key test	1,289
Gibberish test	2,000
Total	6,940

Table 4: Entity statistics of the datasets used to test the connection between distance and familiarity.

4.4 Limitations and Obstacles

Our implementation and experiments run into several limitations and obstacles. To aid future research, this section details these obstacles.

During implementation, we ran into several programming bugs left in the code which affected the output of our model. One bug in the LEAN-LIFE pipeline caused each sentence during prediction with TriggerNER to be reduced to its first character. Another bug caused trigger keys to use the global list of tokens rather than the relevant training sentence when converting key indices to strings. This in turn caused the training to result in significantly less trigger keys and the keys' strings to be nonsensical. These bugs have been patched in section 3.3.2 while modifying TriggerNER. The LEAN-LIFE code with these patched bugs have been made publicly available in section 6.2.

As described in section 4.1, the NLTK tokenization model we use may fail to split multiple tokens due to the versatile grammar found within the dataset of Tweets. Entity triggers may lose their associated named entity. As entity triggers must be associated with a named entity, we opt to cull these entries.

Our original vision for explanations included generating entity triggers within the prediction sentence itself. However, TriggerNER does not natively support this. TriggerNER instead computes the entity trigger most similar to the prediction sentence as approximation for efficiency. We extract the trigger key associated with this entity trigger as an approximation of this explanation as well. Notably, trigger keys are not associated with one specific named entity in a predicted sentence and a predicted sentence may give multiple named entities, so trigger keys do not explain why one specific named entity was tagged. A naive implementation to determine entity triggers within a sentence during prediction was attempted, but this added a significant amount of time to the prediction process and was abandoned due to time constraints. More research may be able to find a less naive and more efficient implementation accomplishing this.

Finally, TriggerNER utilizes randomized shuffles during training which are not properly seeded. This reduces the reproducibility of our experiments. In order to make

the results of our experiments reproducible, we make the trained models used during evaluation publicly available in section 6.2. Future extensions of TriggerNER should implement more deterministic randomization.

5 Results and Analysis

We run experiments using the setup described in section 4 and evaluate the results.

5.1 Overall Performance

In this section, we evaluate the overall performance of the Standard model and the TriggerNER model. We find that the TriggerNER model outperforms the Standard model, achieving a higher F1 score. This increased performance comes from a considerable improvement in the TriggerNER's recall compared to the Standard model, indicating the TriggerNER model achieves less false negatives.

Table 5 displays the performance of the Standard NER model and our TriggerNER model on the verification dataset and the various test datasets. The table has been separated into multiple sub-tables (5a)–(5d) as the datasets have differences worth evaluating separately. For instance, the floods datasets in (5a) and (5b) contain more familiar tokens seen during training than the cyclone dataset in (5c) and the hurricane dataset in (5d). The best-performing scores of each column have been bolded for emphasis.

5a. Floods dev data			
Model	Precision	Recall	F1 score
Standard NER	**98.75**%	69.60%	81.65%
TriggerNER	91.83%	**84.14**%	**87.82**%

5b. Floods test data			
Model	Precision	Recall	F1 score
Standard NER	**96.15**%	52.97%	68.31%
TriggerNER	86.96%	**84.75**%	**85.84**%

5c. Cyclone test data			
Model	Precision	Recall	F1 score
Standard NER	**91.27**%	8.493%	15.54%
TriggerNER	82.60%	**58.57**%	**68.54**%

5d. Hurricane test data			
Model	Precision	Recall	F1 score
Standard NER	**97.44**%	18.68%	31.35%
TriggerNER	75.93%	**49.95**%	**60.26**%

Table 5: Evaluation of the dev and test datasets with and without entity triggers.

The TriggerNER model achieves a higher F1 score than the Standard model on each dataset by 6.2–53%, indicating a more successful model. This performance increase comes from an improvement on the recall score, where TriggerNER outperforms the Standard model on each dataset by 15–50%. This indicates that TriggerNER produces less false negatives, finding more named entities. This improvement comes at a slight cost of precision, as the Standard model achieves a 6.9–22% higher precision score than TriggerNER on each dataset. This indicates that TriggerNER also produces

more false positives, though to a lesser degree than it prevents false negatives at. The significant performance increase on the recall and F1 score on the cyclone and hurricane test datasets indicates the TriggerNER model achieves greater generalizability to unseen entities. Overall, we conclude that the TriggerNER model trained on our trigger-enhanced dataset outperforms the Standard model trained without entity triggers. The TriggerNER model itself as well as our dataset enhanced with entity triggers as per our style guide are considered useful for NER modelling.

5.2 Error Analysis

In this section, we evaluate the errors made by the Standard model and TriggerNER model. We conclude that the TriggerNER model is better at inferring named entities from surrounding context, whereas the Standard model performs better in sentence fragments without context.

We evaluate the errors made by both models on the floods test dataset. In the process, we find that a handful of errors are actually errors in the test dataset's ground truth where the prediction was correct. These errors in the ground truth are detailed in tables 6 and 7. Some prediction errors may also be counted as a correct prediction while the ground truth was not incorrect either. One such example is a named entity "Alappuzha district" which was predicted as just "Alappuzha." We dub these errors "synonymous errors." These synonymous errors are further detailed in table 8. Altogether, 5 errors from the Standard model can be considered correct predictions instead, of which 2 false negatives, 3 false positives and no synonymous errors. The TriggerNER model finds 21 errors which can be considered correct predictions, of which 2 false negatives, 12 false positives and 7 synonymous errors.

We further elaborate on the errors detailed in tables 6 – 8. Table 6's error (1) is annotated incorrectly in the ground truth as the "@" symbol should not be considered part of the location entity. Error (2) is an error in the ground truth as the named entity does not refer to a specific geographical entity.

For table 7, errors (1)–(10) are trivial. The named entities predicted by the NER models are locations, whereas the ground truth did not annotate these as locations. Errors (11) and (12) arise due to tokenization errors on account of grammatical errors in the source text. Both of these errors contain a specific location and exist in a context hinting at a location. However, these tokens also contain characters not part of their respective locations. In case of error (11), the possessive "s" was not separated from the location as its apostrophe is absent. In error (12), the period and "All" were not tokenized separately as the space after the punctuation mark is absent. We consider these as detected locations for completeness. As for error (13), two distinct locations are given within the same token, connected by an underscore. We count this as a valid location as well.

In table 8, errors (1)–(3) identify the specific location mentioned in the ground truth but ignore the generic location words succeeding these. We consider these errors to be valid synonyms of the ground truth's named entity. Errors (4)–(7) are the inverse situation, as both NER models label the generic location words succeeding the specific location's names whereas the ground truth opts not to. We consider these errors valid synonyms as well.

Erroneous false negatives in floods test predictions				
#	Sentence	Ground Truth	Standard NER	TriggerNER
(1)	Did @Cristiano Ronaldo Donate 77 Crore for **@Kerala** Flood? If yes, everyone should #PROUD Of him.	@Kerala	@Kerala	
(2)	#**KeralaFloods Relief Housing**. This technology is suitable for #rehabilitation housing needed for the lakhs of people displaced by the floods. Fast, Solid, Low Cost. #KeralaFloodRelief #affordableHousing #construction	KeralaFloods Relief Housing	—	

Table 6: False negatives in the floods test data predictions coming from annotation errors in the ground truth.

Erroneous false positives in floods test predictions				
#	Sentence	Ground Truth	Standard NER	TriggerNER
(1)	**India** has refused to accept overseas donations for flood relief in Kerala, Thailands Ambassador to India Chutintorn Sam Gongsakdi has said.	—	India	
(2)	Comrade Saji Cherian,our MLA from **Chengannur** constituency who is participating in the flood relief work in Kerala. Our resolve to build New Kerala is getting strengthened by active support of the people.	—	—	Chengannur
(3)	RT @dinesh_rajini: Anyone please help them. #KeralaSOS, #KeralaFloods , #KeralaFloods,#**Kerala** , #KeralaFloodRelief	—	Kerala	—
(4)	Requirement of things in DD Global Village, #**Aluva** - 1. Drinking Water - For 100 Occupants 2. Sanitary Napkins - 20Women 3. Baby Diapers 4. Rice 5. Cereals 6. Grains 7. Sugar 8. Tea Powder 9. Skimmed Milk Powder 10. Atta #KeralaFloodRelief #WeAreKerala #SupportKerala	—	—	Aluva

Continued on the next page

Continued from the previous page

#	Sentence	Ground Truth	Standard NER	TriggerNER
(5)	A Family Need urgent Help. contact number:8113889990, **Chattukulam Siva Temple** Rd, Kadungalloor, Aluva, Kerala 683511 #KeralaFloods	—	—	Chattukulam Siva Temple
(6)	Pamba river near **chengannur** #Thiruvalla before and after kerala floods #KeralaFloods #Keralafloods2018	—	—	chengannur
(7)	**Madhya Pradesh** police donated Rs 1.31 crore to Kerala Chief Ministers distress relief fund. All police personnel of state will donate 1 days salary for same: Madhya Pradesh Director General of Police (ANI) Here's how you can help #KeralaFlood victims:	—	—	Madhya Pradesh
(8)	Madhya Pradesh police donated Rs 1.31 crore to Kerala Chief Ministers distress relief fund. All police personnel of state will donate 1 days salary for same: **Madhya Pradesh** Director General of Police (ANI) Here's how you can help #KeralaFlood victims:	—	—	Madhya Pradesh
(9)	REST IN PEACE: Sgt. Eddison Hermond, a member of the Maryland National Guard and an Air Force veteran, was helping rescue a woman and her cat when the rushing waters swept him away during Sundays flooding in Ellicott City, **MD**	—	—	MD
(10)	RT @CAChirag: Should Central Govt Accept Financial help of 700 Cr from **UAE**? #KeralaFloods #UAE #Kerala	—	—	UAE
(11)	India has refused to accept overseas donations for flood relief in Kerala, **Thailands** Ambassador to India Chutintorn Sam Gongsakdi has said.	—	—	Thailands

Continued on the next page

Continued from the previous page				
#	Sentence	Ground Truth	Standard NER	TriggerNER
(12)	As a relief measure to the people of severely flood hit Kerala, 9 lakhs litres of filtered drinking water sent to kerala by a water special train from Ratlam to **Palghat.All** 15 tanks filled with potable water & quality of water of each tank tested #helpinghand #KeralaFloodRelief	—	—	Palghat.All
(13)	My brothers, aunt and Grand-Parents are stuck in **Moozhikkakadavu_pariyaram**, Chalakkudy. Location - 10.308208,76.351140. Phone - +918075659446. Please RT so some1 can help them. Grand-parents health getting worse. Havent had proper food in 2 days. #KeralaFloods #KeralaSOS	—	Moozhikkakadavu_ pariyaram	

Table 7: False positives in the floods test data coming from annotation errors in the ground truth.

Errors in floods test predictions considered valid synonyms				
#	Sentence	Ground Truth	Standard NER	TriggerNER
(1)	**Pamba river** near chengannur #Thiruvalla before and after kerala floods #KeralaFloods #Keralafloods2018	Pamba river	—	Pamba river
(2)	#WATCH: Exclusive OTV on-ground coverage from one of the largest relief camps set up in a school in Kanichukulangara of **Alappuzha district** #Kerala. The school is currently sheltering around 5,500 people affected in #KeralaFloods #OTVInKerala #OTVExclusive	Alappuzha district	—	Alappuzha district
Continued on the next page				

Continued from the previous page				
#	Sentence	Ground Truth	Standard NER	TriggerNER
(3)	Generous gesture :**Maharashtra State** Road Transport Corporation headed by Shiv Sena leader and transport minister Diwakar Raote donated Rs 10 crore to the CMs Distress fund for #KeralaReliefFund. This is 50% of what Fadnavis announced as state aid to Kerala. #KeralaFloods	Maharashtra State	—	Maharashtra State
(4)	Flash floods ravage Maryland town: A state of emergency was declared in **Howard County**, as a massive storm drenched the Baltimore region, triggering flash floods in Ellicott City and leaving one person missing.	Howard County	—	Howard County
(5)	Please pray for a brave vet who is missing in Maryland in the flooding. **Ellicott City**, Maryland flash floods leave National Guard member missing, devastate town recovering from 2016 deluge #FoxNews	Ellicott City	—	Ellicott City
(6)	Eddison Hermond, a National Guard sergeant who is believed to have been swept away while helping a woman find her cat during **Ellicott City** flooding, is still missing	Ellicott City	—	Ellicott City
(7)	**Howard County** officials are searching for a Maryland National Guardsman reported missing from Sunday's devastating floods. Police say he was helping search for a missing cat when he was last seen.	Howard County	—	Howard County

Table 8: Errors in the floods test data's predictions which may actually be considered valid synonyms of the ground truth's named entities.

Among all errors, we find that the Standard model is better at picking out named entities that have no surrounding context, such as hashtagged locations appended at the end of a Tweet. The Standard model correctly predicted 6 hashtagged locations outside context at the end of a Tweet which the TriggerNER model did not predict. This includes 1 erroneous false positive from table 7. The TriggerNER model has not found any locations like this which were missed by the Standard model.

TriggerNER meanwhile excels at picking out named entities within context, such as locations following the word "in." TriggerNER predicted 95 of these contextual entities

which the Standard model did not detect, including 10 erroneous false positives from table 7. The Standard model picked out 1 contextual entity missed by TriggerNER.

As a result of TriggerNER's increased contextual generalizability, a few more errors made by TriggerNER are false positives for words that follow context which resembles context surrounding locations, such as "backdrop" in "they might cancel it in backdrop" and "ur" in "teams active in ur location." TriggerNER predicted a total of 10 false positives like these while the Standard model predicted 1 false positive like this. These exclude the erroneous false positives from table 7 and the synonymous errors from table 8.

5.3 Distances

In this section, we show that the distances given by the model are not a measure of confidence. Instead, distances can be used as a measure of familiarity. In particular, a distance tells us how similar the tokens closest to the outer edges of the sentence are to the nearest trained trigger in the eyes of the model.

Each prediction made by our TriggerNER model also outputs the numerical value d, which is the distance between the encoded prediction sentence vector and the nearest encoded trigger vector learned during training. This distance is computed for the entire prediction sentence and not any specific token or predicted named entity. Our hypothesis states that this distance can be utilized as a measure of confidence in the produced entity prediction. In order to test this hypothesis, we first evaluate the ranges, means and medians of each dataset, categorized by predictions which were correct and incorrect. The results are given in table 9. Table 9 has been separated into multiple sub-tables (5a)–(5c) in order to evaluate differences between datasets containing more familiar tokens seen during training and less familiar tokens.

9a. Floods test data

Data	Count	Smallest distance	Greatest distance	Mean	Median
All data	162	0.007318	1.539	0.5626	0.5307
Correct predictions	117	0.009083	1.532	0.5673	0.5147
Predictions with errors	45	0.007318	1.539	0.5506	0.5565

9b. Cyclone test data

Data	Count	Smallest distance	Greatest distance	Mean	Median
All data	1,038	0.01013	1.950	0.5829	0.5293
Correct predictions	553	0.01013	1.565	0.5856	0.5326
Predictions with errors	485	0.08894	1.950	0.5799	0.5203

9c. Hurricane test data

Data	Count	Smallest distance	Greatest distance	Mean	Median
All data	1,038	0.09380	1.577	0.5985	0.5394
Correct predictions	599	0.1024	1.577	0.5998	0.5398
Predictions with errors	539	0.09380	1.516	0.5967	0.5391

Table 9: Ranges and averages of prediction trigger distances found for the test datasets.

The results show no significant difference between distances found for correct and incorrect predictions. Each dataset achieves a mean distance close to 0.59 and a median

close to 0.53 regardless of whether correct or incorrect predictions are evaluated. We note that the floods test dataset finds smaller lower bounds to its range and a lower mean average, despite having less samples to achieve a lower bound with. As shown in figure 1, the floods test dataset results in a notably higher proportion of predicted distances below 0.1 than the other two test datasets. This may indicate a relation between lower distances and familiarity, as the floods test data resembles the training data more closely.

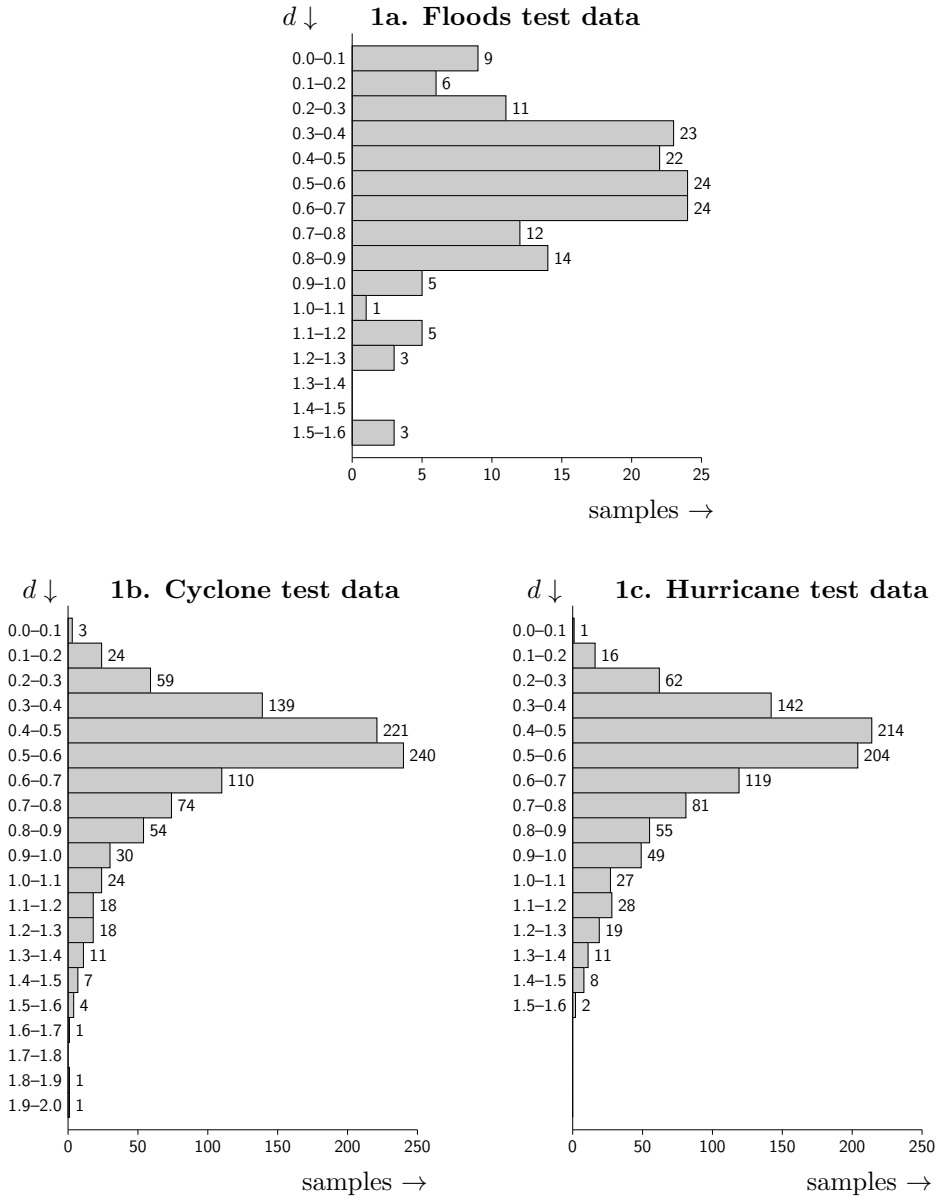

Figure 1: Frequency of prediction trigger distances in the test datasets.

In addition, we analyze the obtained range of distances between correct and incorrect predictions using the Mann-Whitney U test. This helps determine whether the average distances for correct and incorrect predictions are truly not statistically significant. The Mann-Whitney U test can be performed as each prediction is independent and the

distribution between correct and incorrect predictions is similar. A small p-value such as 0.05 or lower would indicate a statistical significance between the distribution of distances of correct and incorrect predictions. However, the floods test data achieves a p-value of 0.90, cyclone test achieves 0.67 and hurricane test achieves 0.79. We conclude that there is no significant difference between the distances of correct and incorrect predictions.

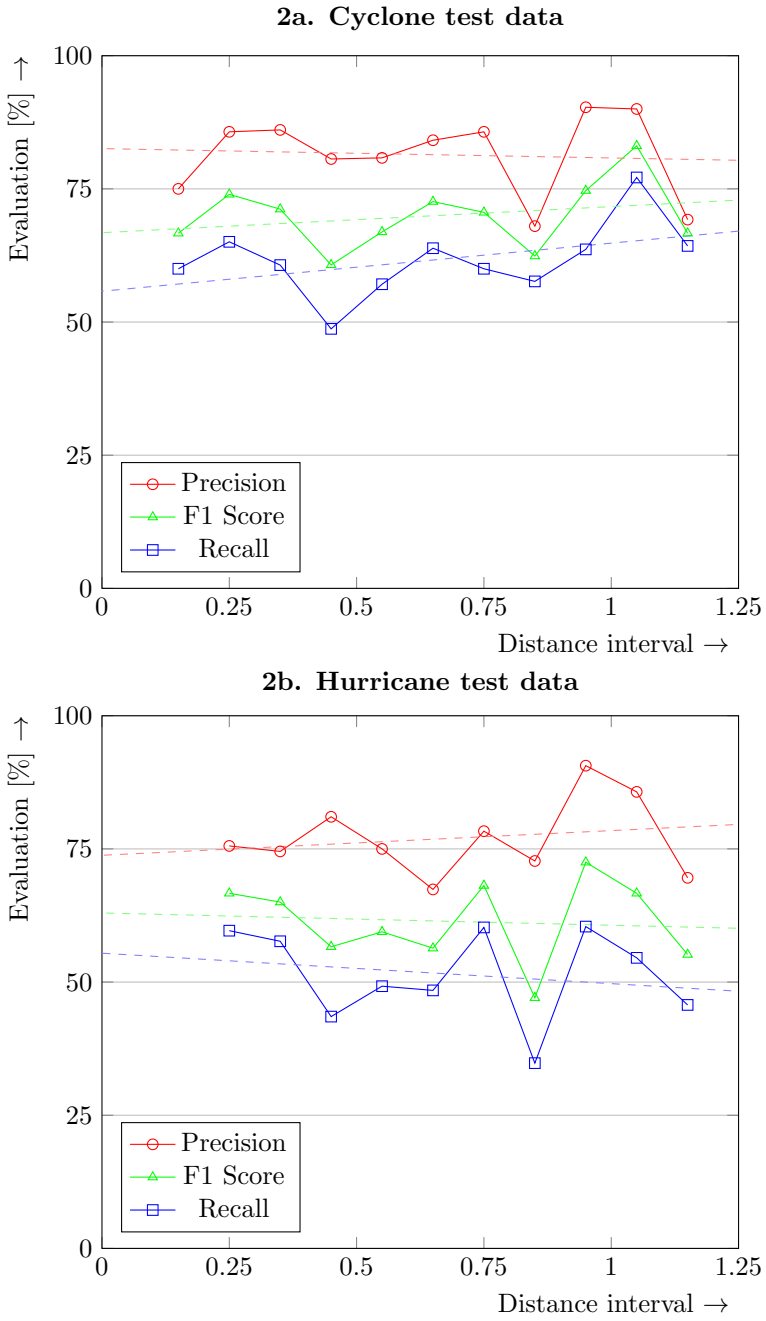

Figure 2: Evaluation of the cyclone and hurricane test datasets ordered by prediction trigger distance with intervals occurring less than 20 times culled.

Next we evaluate the accuracy of the model's predictions more closely across smaller intervals of distance in order to determine whether smaller distances provide greater accuracy and subsequently greater confidence. The amount of predictions made for each distance interval can be found in figure 1. These performance scores of each interval for the larger two datasets are plotted in figure 2. Intervals containing less than 20 predictions are culled from figure 2 in order to ensure a decent sample size for each data point while determining a pattern.

The performance patterns seen in figures 2 show no significant correlation between distances and performance either. We reject the hypothesis stating that the distance offers a measure of confidence of the prediction's accuracy.

We develop a new hypothesis. We hypothesize that the distance value offers a measure of familiarity. In other words, prediction sentences which resemble those seen during training are expected to produce lower distance values than prediction sentences built from unseen tokens. We use the datasets detailed in table 4. These datasets are constructed as follows:

- Training test: 1,742 unique Tweets consisting of all training data which features entity triggers.
- Entity trigger test: 1,909 unique entity triggers taken from the training data.
- Trigger key test: 1,289 unique trigger keys, composed from the final list of trigger keys obtained during training.
- Gibberish test: 2,000 unique sentences consisting of randomly generated alphanumerical noise.

We define hypothesis (i) stating that the entity trigger test and trigger key test datasets will achieve lower distances than the averages of around 0.53–0.59 found in table 9, as these datasets contain entries closely resembling trained triggers. In addition, we define hypothesis (ii) stating the gibberish test dataset will achieve greater distances than average as its entries will generally not resemble those found in the training data. Finally, we define hypothesis (iii) stating that the training test dataset will achieve lower distances than average but greater than those found for the entity trigger test and trigger key test datasets, as the tokens found in this dataset are familiar training data but not as similar to the trained trigger keys. The ranges and averages we find for each dataset are given in table 10.

Distance familiarity test datasets				
Dataset	Smallest distance	Greatest distance	Mean	Median
Training test	0.001613	1.799	0.3228	0.01958
Entity trigger test	0.1890	1.922	0.9324	0.9364
Trigger key test	0.2196	1.839	0.9166	0.9285
Gibberish test	0.2026	1.593	0.5365	0.5144

Table 10: Ranges and averages of prediction trigger distances found for familiarity tests.

The results of table 10 indicate a degree of familiarity with the training data affecting the distances.

The test data identical to the trigger-annotated training data achieve much lower average distances than the other test datasets in tables 9 and 10, including the entity trigger test and trigger key datasets. Therefore, we reject hypothesis (iii). Additionally, the test data containing entity triggers and trigger keys do achieve similar distances

to each other, but much higher than the expected average of 0.53–0.59. We reject hypothesis (i). Finally, the gibberish test data attains fairly average distances also seen in table 9, rather than high distances. We reject hypothesis (ii). The notably low distances attained by the training test dataset indicates that low distances indicate similarity to the training data. However, the high distances obtained by the entity trigger test and trigger key test datasets indicate that the training test dataset's low distances are not due to the similarity to the trained entity triggers.

We develop another hypothesis. As the gibberish test data consists of alphanumerical noise, no punctuation symbols are found in it. If we place a period at the end of each gibberish sentence, the gibberish data will more closely resemble the training data which often has sentences ending in a period as well. Thus we hypothesize that the distances found for the gibberish test dataset will lower if we append a period token at the end of each prediction sentence. We find that the attained distances do indeed lower. The smallest distance is 0.17, the greatest distance is 1.0, the mean is 0.46 and the median is 0.46.

We hypothesize that all prediction distances move closer to a specific value when each sentence ends in a period, again to make sentences resemble the training data. We modify each test dataset by adding an additional period token at the end of each sentence. We find that the mean for every modified test dataset is between 0.46–0.58 and the median is between 0.46–0.59. Experimenting further by placing a period token at different parts of the sentence, we find that a similar effect occurs when prefixing each sentence with a period token. Placing a period at the start and end of each sentence produces a mean distance across all datasets of 0.36–0.50 and a median of 0.36–0.51.

We test the effect of tokens in the first or last position of the sentence further by using different tokens than a period. We find that each token is associated with particular distance values they are inclined to move the prediction towards. We exhibit this by modifying all test datasets such that each prediction moves towards a very low value. We also show that we can move each prediction to a very high value. First, we take each of the preceding test datasets (floods test, cyclone test, hurricane test, training test, entity trigger test, trigger key test and gibberish test). We look up which prediction sentences achieved the smallest and greatest distance across all datasets and take note of their first and last tokens, in this case respectively "Marylands" and "tictocnews" for the smallest distance (0.001613) and "Flooding" and ">" for the largest distance (1.950). All test datasets are modified by adjoining the tokens associated with the smallest distance to the first and last positions of each sentence. Each modified dataset is evaluated on achieved distances again and the results are given in table 11. The same evaluation is performed by modifying all test datasets with the first and last tokens associated with the greatest distance, with results given in table 12.

The results of these tests given in tables 11 and 12 show that the distance values are strongly affected by the first and last token of a sentence. We show that any sentence can be strongly coerced to a particular distance value by prefixing or suffixing a particular token to it.

“Marylands [. . .] tictocnews” familiarity datasets				
Dataset	Smallest distance	Greatest distance	Mean	Median
Modified floods test	0.03789	0.08386	0.05571	0.05477
Modified cyclone test	0.03523	0.1156	0.05890	0.05834
Modified hurricane test	0.03525	0.1064	0.05691	0.05588
Modified training test	0.01869	0.2715	0.06488	0.05537
Modified entity trigger test	0.01943	0.1125	0.04310	0.04094
Modified trigger key test	0.01943	0.1216	0.04422	0.04269
Modified gibberish test	0.03064	0.1658	0.04670	0.04555

Table 11: Ranges and averages of prediction trigger distances found by adjoining the tokens “Marylands” and “tictocnews” to the first and last respective position of each sentence.

“Flooding [. . .] >” familiarity datasets				
Dataset	Smallest distance	Greatest distance	Mean	Median
Modified floods test	1.657	1.778	1.680	1.679
Modified cyclone test	1.645	1.780	1.682	1.676
Modified hurricane test	1.652	1.779	1.681	1.679
Modified training test	0.8370	2.041	1.814	1.868
Modified entity trigger test	0.8137	2.030	1.671	1.734
Modified trigger key test	0.7446	2.030	1.656	1.728
Modified gibberish test	1.395	2.015	1.786	1.784

Table 12: Ranges and averages of prediction trigger distances found by adjoining the tokens “Flooding” and “>” to the first and last respective position of each sentence.

Distances against quantity of predictions without NEs						
Interval	Cyclone test			Hurricane test		
	With NEs	Without NEs	Total	With NEs	Without NEs	Total
0.0–0.1	1	2	3	1	0	1
0.1–0.2	11	13	24	4	12	16
0.2–0.3	33	26	59	29	33	62
0.3–0.4	76	62	138	67	75	142
0.4–0.5	107	114	221	87	127	214
0.5–0.6	132	108	240	95	109	204
0.6–0.7	74	36	110	63	56	119
0.7–0.8	46	28	74	32	49	81
0.8–0.9	31	23	54	17	38	55
0.9–1.0	18	12	30	18	31	49
1.0–1.1	17	7	24	9	18	27
1.1–1.2	10	8	18	12	16	28
1.2–1.3	13	5	18	10	9	19
1.3–1.4	4	7	11	4	7	11
1.4–1.5	2	5	7	1	7	8
1.5–1.6	3	1	4	1	1	2
1.6–1.7	0	1	1	0	0	0

Table 13: Distance intervals 0.0–1.7 with the corresponding quantity of predictions with feature at least one named entity or zero named entities for the cyclone test and hurricane test datasets.

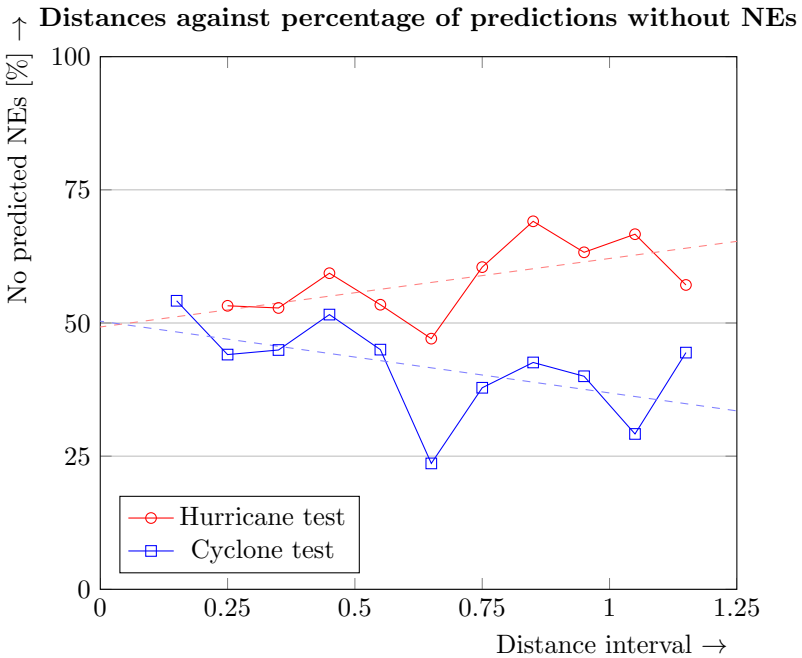

Figure 3: Distance intervals plotted against percentage of predictions without named entities for the cyclone test and hurricane test datasets with intervals occurring less than 20 times culled.

In addition, we test whether distances correlate to a tendency to predict no named entities. Each predicted distance is categorized on intervals of size 0.1. We sum the amount of predictions which feature zero named entities by their associated distance interval in table 13. We statistically test if there is a correlation between the achieved distance interval and whether or not a named entity was predicted by performing a chi-squared test over the distance range 0.0–1.7 for the cyclone test and hurricane test datasets. We define a null hypothesis stating that the distance intervals and quantity of predictions with or without named entities are independent. A small p-value such as 0.05 or lower may disprove this null hypothesis for a dataset. We compute the test statistic for both test datasets using the values found in table 13. The test statistic for the cyclone test dataset sums to 23.73 and the test statistic for the hurricane test dataset sums to 20.55. The associated chi-squared distribution of these sums are respectively 16 and 15. Thus we compute a p-value for the cyclone test dataset of 0.023 and a p-value for the hurricane test dataset of 0.035. As these p-values are both below 0.05, we reject the null hypothesis and conclude that there is a correlation between the distance intervals and the quantity of predictions with or without named entities.

We further examine this correlation by plotting the found distance intervals against the percentage of predictions which feature no named entities. We only plot distance intervals with at least 20 predictions in order to assure a decent sample size for each data point when determining a pattern. See figure 3. The percentage of named entities predicted for each data interval varies sporadically between intervals. There is no common pattern between the two datasets. Furthermore, the trend line for the cyclone test dataset slowly decreases in the amount of predictions without named entities as distance increases whereas the hurricane test dataset's trend line increases. We conclude that while the chi-squared test shows there exists a correlation between the distance interval and whether or not named entities are predicted, it does not generalize to a consistent relationship between datasets. This makes distances untenable as explanation as to why no named entities may be predicted for a sentence.

From the results of tables 10, 11 and 12, we conclude that distances express the familiarity between the training data and the bordering tokens of prediction sentences. In the case of our trained models, a value below 0.53–0.59 indicates greater familiarity while a value above 0.53–0.59 indicates greater unfamiliarity. Models trained on other entity trigger data achieve a different trigger vector space and would find a different threshold. No correlation is found between distances and accuracy. There may be a correlation between distances and the prediction of zero named entities, but this does not generalize to an effective explanation.

The familiarity being as dependent as it is on the bordering tokens of a prediction sentence indicate a possible concern. For instance, a period inserted at the end of a sentence would not affect the familiarity of a sentence in drastic ways to a human reader. However, the distance of all sentences beginning or ending with a period token is strongly coerced towards a specific value just below average. In addition, table 11 shows that any sentence can seem incredibly familiar just by including the token "Marylands" at the start and the token "tictocnews" at the end. This includes even sentences which are complete gibberish. Similarly, table 12 shows that even training data can seem wholly unfamiliar when the tokens "Flooding" and "¿" are appending to the ends of its sentences. This behavior differs severely from how humans would interpret sentences. The behavior may be indicative of an erroneous pattern of recognition having been learned by the model.

5.4 Trigger Keys

In this section, we evaluate how applicable trigger keys are as explanations. We find that the string of text provided as a trigger key is not useful for human interpretation.

Our extended TriggerNER model outputs a trigger key k_t with each prediction. Trigger keys are predicted for the entire prediction sentence and not any particular token or predicted named entity. This trigger key gives a human-readable representation of the trained entity trigger whose encoding is deemed most similar to the encoded prediction sentence. These most similar entity triggers from the training data are used by the model as an approximation of the entity triggers in the prediction sentence without requiring the model to pick out entity triggers for each named entity in the prediction sentence itself. In this section, we analyze how useful these trigger keys are as explanations to the given predicted named entities. We also analyze whether there is a correlation between trigger keys' attained distances, performance and amount of sentences predicted without named entities.

The TriggerNER model trained on our dataset learns a total of 1,289 trigger keys. During prediction, a total of 613 different trigger keys are found across all test datasets. Of these, 134 are found in the floods test data, 398 are found in the cyclone test data and 403 are found in the hurricane test data. The 20 most common trigger keys across all three test datasets are found in table 14. Each trigger key is given with its respective occurrence count, distance range and averages, performance evaluation and percentage of sentences predicted to have no named entities.

The small sample size for each trigger key makes it difficult to determine any strong relationship between trigger keys and distances, performance or predictions without named entities. Analyzing the relationship between trigger keys across table 14 which attain lower or higher distances, performances and predictions without named entities reveals no obvious correlation between the three statistics among the most common trigger keys either. The table shows that higher distances occur similarly between higher or lower performances, higher performances occur between higher or lower percentages of predictions without named entities, and so on. As an example, the trigger key "aid for" achieves an average distance of 0.50 and one of the lowest observed performances with an F1 score at 44%. However, the trigger key "Center" achieves a very similar average distance of 0.51–0.52 and one of the highest observed F1 scores at 67%. No patterns are identified.

We analyze the interpretability of the string of text given as the trigger key for each prediction. A total of 613 out of 1,289 trigger keys occur in our predictions. One example of a prediction:

Khalsa Aid to the forefront!!

$$k_t = \text{set up in}$$
$$d = 0.7706$$

Trigger keys function as a label for which specific trained trigger vector was picked as closest approximation of the prediction sentence, much like a form of identification. As explanation for the decision itself, however, trigger keys offer little insight. Tokens found in the trigger key are rarely found in the prediction sentence. In the previous example, none of the tokens in the trigger key "set up in" appear in the sentence. Even when the prediction sentence features tokens found in the trained trigger keys, a less-related trigger key is often picked. For instance, the previous example's trigger key "set up in" appears in the following sentence:

#WATCH: Exclusive OTV on-ground coverage from one of the largest relief camps

set up in a school in Kanichukulangara (LOC) of Alappuzha (LOC) district #Kerala (LOC). The school is currently sheltering around 5,500 people affected in #KeralaFloods #OTVInKerala

#OTVExclusive

k_t = at EDAYARANMULA

$d = 0.5565$

Although this prediction sentence includes the exact phrase "set up in," the aforementioned trigger key "set up in" was not determined to be the closest approximation of this sentence. Instead the trigger key "at EDAYARANMULA" is the closest approximation, despite having no tokens in common with the sentence. Any semantic relatedness between the trigger key and sentence is difficult to determine.

Even when there is a trigger key that is a perfect match to a prediction sentence, the closest approximation of the sentence is likely still an unrelated trigger key. Take the following example:

People from South India (LOC)

k_t = counties in Maryland

$d = 0.9478$

The trigger key "counties in Maryland" is a perfect lexical and semantic match with the identical sentence:

counties in Maryland (LOC)

k_t = PARTS OF SOUTHERN

$d = 1.026$

Yet its closest trigger key is determined to be a phrase with no tokens in common. Ambiguous decisions like this are found repeatedly throughout the test data results. Ultimately, we conclude that trigger keys offer little human insight to the decision of which trained trigger is the best approximation for a sentence, nor which token groupings are chosen as named entities.

The mystifying choice of trigger key may indicate erroneous reasoning from the model. As shown, the model often picks trigger keys with no lexical or semantic similarity to the prediction sentence even when more suitable trigger keys are available. Human users would opt for notably different trigger keys than the model would. This may indicate that the model has learned undesirable reasoning during its training.

Trigger keys									
Trigger key	Total count	Smallest distance	Greatest distance	Mean	Median	Precision	Recall	F1 Score	Predictions without NEs
aid for	62	0.3566	0.6641	0.4975	0.4991	66.67%	32.26%	43.48%	62.90%
Center	51	0.2686	0.7943	0.5179	0.5127	80.00%	57.14%	66.67%	39.22%
surrounding	47	0.1929	0.8392	0.4841	0.4757	72.92%	53.85%	61.95%	40.43%
has been send off to	35	0.3808	0.8776	0.5031	0.4842	63.64%	37.84%	47.46%	54.29%
destroyed state	29	0.3148	0.6142	0.4855	0.5217	66.67%	37.04%	47.62%	58.62%
flooding in parts of eastern	25	0.2089	0.7125	0.4401	0.4088	78.57%	47.83%	59.46%	56.00%
ChildFund	23	0.1308	0.8186	0.4128	0.3880	73.33%	61.11%	66.67%	52.17%
in flood affected regions of	23	0.4760	1.007	0.5964	0.5369	67.86%	47.50%	55.88%	47.83%
HADR operations in	20	0.1202	0.5354	0.3651	0.3746	85.71%	63.16%	72.73%	45.00%
Maldives	20	0.1781	0.6813	0.3783	0.3538	50.00%	33.33%	40.00%	75.00%
Missing from	20	0.3082	0.5472	0.4313	0.4269	85.71%	42.86%	57.14%	65.00%
People	18	0.4074	1.077	0.5930	0.5633	94.12%	57.14%	71.11%	38.89%
Kodagu flood victims	17	0.3853	0.6965	0.5328	0.5286	83.33%	71.43%	76.92%	29.41%
Air National Guard	16	0.4680	0.8912	0.5665	0.5322	76.92%	58.82%	66.67%	56.25%
parts of flood affected areas in	16	0.5059	0.9501	0.7241	0.7102	100.0%	50.00%	66.67%	75.00%
group	15	0.3555	0.7152	0.4785	0.4829	66.67%	47.62%	55.56%	40.00%
is committed to support	15	0.4018	0.6190	0.5309	0.5545	81.82%	81.82%	81.82%	40.00%
fund Relief	15	0.4454	0.7285	0.5761	0.5537	71.43%	23.81%	35.71%	60.00%
hands over	15	0.1524	0.9131	0.3222	0.2331	100.0%	72.22%	83.87%	46.67%
area surroundings of	14	0.3294	0.5696	0.4223	0.4240	90.91%	66.67%	76.92%	42.86%

Table 14: Occurrence count, distance ranges and averages, performance and amount of sentences predicted without named entities corresponding to the 20 most common trigger keys found across the floods test, cyclone test and hurricane test datasets.

6 Conclusion and Outlook

6.1 Conclusion

In this paper, we test the applicability of entity triggers in XAI.

By extracting information about the decisions made by the TriggerNER model during the NER prediction process, more insight is gained into the process. The explanations which can be extracted from the TriggerNER model consist of trigger keys and distances. Trigger keys and distances relate to the entire prediction sentence rather than particular tokens or predicted named entities. Trigger keys are strings of text representing trigger vectors, aiding human interpretability. Distances indicate how similar the prediction sentence is to the training data in the eyes of the model. Therefore, the answer we reach to the second sub-question formulated in section 1.2, "What explanations are offered to justify tagging an entire Tweet as containing a location entity?" is trigger keys and distances.

The trigger keys describe text from the training data rather than the prediction data. It is generally difficult to see the connection between the trigger key and the prediction sentence. Trigger keys ultimately do not offer a cogent nor intuitive explanation for prediction decisions. The distance value gives a sense of how similar the model considers the prediction sentence to be to the predicted entity trigger. We find that this distance has a focus on the first and last token of the sentence. We also find that this distance is lower when these bordering tokens are similar to the border tokens found in training data. The TriggerNER model trained on our training data achieves an average distance of around 0.53–0.59, where distances below this average are considered more familiar and larger distances are considered less familiar. This achieved average distance depends on the entity trigger vector space learned from the training data and a model trained on different training data will achieve a different range of distances. However, no connection is found between distance size and performance of the model. Altogether, entity triggers do not offer much additional explanation towards prediction choices or performance. The third sub-question "Are the offered explanations intuitive?" is answered in the negative.

Trigger keys and distances offer little correlation to interpret why the model may determine there are no named entities in a sentence. The answer to our fourth sub-question, "What explanations are offered to justify tagging an entire Tweet as **not** containing a location entity?" is none.

As part of this paper, we extended a floods disaster dataset by annotating entity triggers in accordance to a style guide we composed. The overall performance of the NER model improved with the inclusion of these entity triggers, boasting a higher F1 score by 6.2–53% and much higher recall score by 15–50%. False positive predictions were slightly more common by 6.9–22%. The trigger-enhanced model improves the generalizability of predictions. This answers the first sub-question, "What is the difference in performance between algorithms trained on Tweet data annotated with entity triggers and trained without entity triggers?"

Although overall performance and generalizability improves with the use of entity triggers, the explanations found to be offered by the model are unsatisfactory. The chosen trigger key often has no tokens in common with the sentence. Any semantic relation is often distant. Distances could offer some explanation by determining that a prediction sentence is similar to data seen before during training. However, the strong weighting effected by the first and last tokens get in the way of this utility. For instance, we find that placing a period at the start or end of the sentence reduces the distance below average. In addition, we find that total gibberish is considered very familiar if it respectively starts and ends with one specific token. Likewise, even sentences taken directly from the training data are considered completely unfamiliar if they respectively

start and end with a specific token. These results for trigger keys and distances alike are notably different from how humans would evaluate sentences. As such, it may be that the trigger-enhanced model has learned unsound reasoning in order to approximate entity triggers.

In conclusion, we answer our main research question "How can entity triggers add value as explanations for named entity recognition in disaster risk management?" We extract the string representation of the entity trigger considered to be the most similar to the prediction sentence, as well as a heuristic telling us how familiar the model considers the sentence to be. These are the trigger key and distance, respectively. The trigger key offers little value as an explanation for predicted named entities, having no obvious relevance to the prediction sentence. Distances explain how familiar the model considers a sentence to be from its training data, with a focus on the bordering tokens. This adds some value in the interpretation of an unseen sentence's familiarity. Altogether, not much value is added by entity triggers in the area of user explanation.

6.2 Research Products

We present the style guide written in section 3.1, the trigger-annotated flood Tweets dataset from section 3.2.2, the adjusted TriggerNER model from section 3.3.2, the Jupyter notebook, the trained NER models and all code used throughout this paper at the following GitHub page:

`https://github.com/Yoriyari/Bachelor-Thesis-Trigger-Explanations/`

6.3 Future Directions

The results of this paper offer new directions of research to explore. We find that explanations for the trigger-enhanced NER process remain lacking. Possible avenues to discover new explanations are offered. In addition, we opt that further research into the reasoning done under the hood by the trigger-enhanced NER model is needed. Finally, we recommend ways in which resources for trigger-enhanced NER may be improved.

6.3.1 Explanations Using Entity Triggers

More research must be done on producing interpretable explanations for the prediction process. Currently, TriggerNER approximates the entity triggers present in the prediction sentence by taking the trigger vector obtained during the training process which is interpreted as the most similar to the prediction sentence as per the vector distances. The ability to pick out entity triggers within the prediction sentence itself would offer a more intuitive explanation for human readers. One possible avenue of research may be to test if a NER model can be trained to efficiently predict entity triggers in the prediction sentence. This research might train NER models using entity triggers as its named entities. Each model would only train on entity triggers associated with one specific class. With these models trained to pick out entity triggers from a prediction sentence, research should test whether the results produced by these models perform comparably to the approximations which TriggerNER uses. If they do, this approach may be useful both in terms of accuracy and explainability.

The annotation style guide created for this paper produced a trigger-enhanced dataset which improved the overall performance of the NER model. This indicates it is suitable as a guideline for trigger annotation. However, this style guide is written based on the intuition of the author. Further research will be able to produce trigger annotation semantics which are more comprehensive and may lead to better model performance.

6.3.2 Reasoning of Entity Triggers

We have found that trigger keys and distances provide unreasonable explanations. These may indicate that the model has learned faulty reasoning during its training process. Further investigation is warranted in regard to how the model computes trigger keys and distances. Likely the optimal avenue to explore is the mathematical conversion of entity trigger tokens and prediction sentence tokens into their respective vector representations. As even prediction sentences identical to trained entity triggers result in different predicted entity triggers, the difference in their method of calculation is presumed to be the primary reason trigger keys and distances are not intuitive to human interpretation.

6.3.3 Resources for Entity Triggers

The additional effort involved in annotating entity triggers in datasets, particularly large datasets, may serve to dissuade the adoption of entity triggers in NER. It would be fruitful to reduce the human effort in annotating entity triggers. One approach already being explored by Lee et al. proposes training a model to classify entity triggers automatically and using human-in-the-loop feedback to eliminate incorrect predictions, streamlining the intermediate process [18]. Lee et al. has also proposed a framework streamlining the debugging of models with human-in-the-loop techniques [19].

One additional direction to be explored is generalizing NER models for DRM further. This paper explores English Tweets, but there are many non-English users on Twitter as well. These languages often have significantly less training data available and may not extend as effectively to the NER techniques we use, such as tokenization on whitespaces as not every language uses space-separation. The collection of datasets from different regions and languages as well as the exploration of TriggerNER's utilizability in other languages should be encouraged.

References

[1] B. Y. Lin, D.-H. Lee, M. Shen, R. Moreno, X. Huang, P. Shiralkar, and X. Ren, "TriggerNER: Learning with entity triggers as explanations for named entity recognition," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8503–8511, 2020.

[2] H. Ritchie and P. Rosado, "Natural disasters," 2022. https://ourworldindata.org/natural-disasters (accessed 14 Dec. 2023).

[3] W. Sun, P. Bocchini, and B. D. Davison, "Applications of artificial intelligence for disaster management," *Natural Hazards*, vol. 103, pp. 2631–2689, Sept. 2020.

[4] M. Imran, P. Mitra, and C. Castillo, "Twitter as a lifeline: Human-annotated Twitter corpora for NLP of crisis-related messages," *Proceedings ofthe Tenth International Conference on Language Resources and Evaluation*, pp. 1638–1643, May 2016.

[5] M. Imran, F. Ofli, D. Caragea, and A. Torralba, "Using AI and social media multimodal content for disaster response and management: Opportunities, challenges, and future directions," Sept. 2020.

[6] G. Cervone, E. Sava, Q. Huang, E. Schnebele, J. Harrison, and N. Waters, "Using Twitter for tasking remote-sensing data collection and damage assessment: 2013 Boulder flood case study," *International Journal of Remote Sensing*, vol. 37, pp. 100–124, Jan. 2016.

[7] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticæ Investigationes*, vol. 30, pp. 3–26, Jan. 2007.

[8] R. Suwaileh, T. Elsayed, and M. Imran, "IDRISI-RE: A generalizable dataset with benchmarks for location mention recognition on disaster tweets," *Information Processing and Management*, vol. 60, May 2023.

[9] S. Alqaaidi and E. Bozorgi, "A survey on recent named entity recognition and relation classification methods with focus on few-shot learning approaches," *arXiv*, Oct. 2023.

[10] C. M. Gevaert, M. Carman, B. Rosman, Y. Georgiadou, and R. Soden, "Fairness and accountability of AI in disaster risk management: Opportunities and challenges," Nov. 2021.

[11] J. dela Cruz, I. Hendrickx, and M. Larson, "Towards XAI for information extraction on online media data for disaster risk management," *Proceedings of the 20th ISCRAM Conference*, May 2023.

[12] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An overview of interpretability of machine learning," *IEEE 5th International Conference on data science and advanced analytics*, May 2018.

[13] H. Nakayama, K. Takahiro, J. Kamura, Y. Taniguchi, and X. Liang, "doccano: Text annotation tool for human," 2018. https://github.com/doccano/doccano (accessed 12 Oct. 2023).

[14] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 1064–1074, 2016.

[15] Z. Lin, M. Feng, C. N. D. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2017.

[16] D.-H. Lee, R. Khanna, B. Y. Lin, S. Lee, Q. Ye, E. Boschee, L. Neves, and X. Ren, "LEAN-LIFE: A label-efficient annotation framework towards learning from explanation," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 372–379, 2020.

[17] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. O'Reilly Media Inc., Jan. 2009.

[18] D.-H. Lee, R. K. Selvam, S. M. Sarwar, B. Y. Lin, F. Morstatter, J. Pujara, E. Boschee, J. Allan, and X. Ren, "AutoTriggER: Label-efficient and robust named entity recognition with auxiliary trigger extraction," *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 3011–3025, Sept. 2021.

[19] D.-H. Lee, A. Kadakia, B. Joshi, A. Chan, Z. Liu, K. Narahari, T. Shibuya, R. Mitani, T. Sekiya, J. Pujara, and X. Ren, "XMD: An end-to-end framework for interactive explanation-based debugging of NLP models," *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 264–273, Oct. 2022.