

Data And Knowledge Driven Named Entity Recognition for Cyber Security

Sanjeevini S. H¹, B. Mrudulanjali², T. Pavani², V. Sowmya², T. Shivani²

¹Assistant Professor,² UG Scholar, ^{1,2} Department of CSE-Cyber Security

^{1,2}Malla Reddy Engineering College for Women (A), Maisammaguda, Medchal, Telangana.

Abstract

Named entity recognition (NER) is the task to identify mentions of rigid designators from text belonging to predefined semantic types such as person, location, organization etc. NER always serves as the foundation for many natural language applications such as question answering, text summarization, and machine translation. Early NER systems got a huge success in achieving good performance with the cost of human engineering in designing domain-specific features and rules. In recent years, deep learning, empowered by continuous real-valued vector representations and semantic composition through nonlinear processing, has been employed in NER systems, yielding state-of-the-art performance. In this paper, we provide a comprehensive review on existing deep learning techniques for NER. We first introduce NER resources, including tagged NER corpora and off-the-shelf NER tools. Then, we systematically categorize existing works based on a taxonomy along three axes: distributed representations for input, context encoder, and tag decoder. Next, we survey the most representative methods for recent applied techniques of deep learning in new NER problem settings and applications. Finally, we present readers with the challenges faced by NER systems and outline future directions in this area.

Keywords: Name entity recognition, Cyber security, Deep learning.

1. Introduction

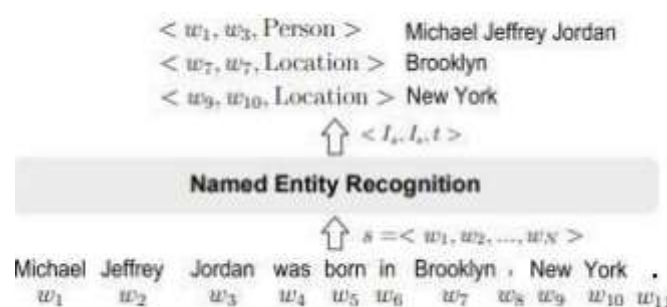
Named Entity Recognition (NER) aims to recognize mentions of rigid designators from text belonging to predefined semantic types such as person, location, organization etc. NER not only acts as a standalone tool for information extraction (IE), but also plays an essential role in a variety of natural language processing (NLP) applications such as text understanding, information retrieval, automatic text summarization, question answering, machine translation, and knowledge base construction etc. Evolution of NER. The term “Named Entity” (NE) was first used at the sixth Message Understanding Conference, as the task of identifying names of organizations, people and geographic locations in text, as well as currency, time and percentage expressions. Since MUC6 there has been increasing interest in NER, and various scientific events (e.g., CoNLL03, ACE, IREX, and TREC Entity Track) devote much effort to this topic. Regarding the problem definition, Petasis et al. restricted the definition of named entities: “A NE is a proper noun, serving as a name for something or someone”. This restriction is justified by the significant percentage of proper nouns present in a corpus. Nadeau and Sekine claimed that the word “Named” restricted the task to only those entities for which one or many rigid designators stands for the referent. Rigid designator, defined in [1], include proper names and natural kind terms like biological species and substances. Despite the various definitions of NEs, researchers have reached common consensus on the types of NEs to recognize. We generally divide NEs into two categories: generic NEs (e.g., person and location) and domainspecific NEs (e.g., proteins, enzymes, and genes).

In this paper, we mainly focus on generic NEs in English language. We do not claim this article to be exhaustive or representative of all NER works on all languages. As to the techniques applied in NER, there are four main streams: 1) Rule-based approaches, which do not need annotated data as they rely

on hand-crafted rules; 2) Unsupervised learning approaches, which rely on unsupervised algorithms without hand-labeled training examples; 3) Feature-based supervised learning approaches, which rely on supervised learning algorithms with careful feature engineering; 4) Deep-learning based approaches, which automatically discover representations needed for the classification and/or detection from raw input in an end-to-end manner. Motivations for conducting this survey. In recent years, deep Learning (DL, also named deep neural network) has attracted significant attention due to its success in various domains. Starting with Collobert et al., DL-based NER systems with minimal feature engineering have been flourishing. Over the past few years, a considerable number of studies have applied deep learning to NER and successively advanced the state-of-the-art performance. This trend motivates us to conduct a survey to report the current status of deep learning techniques in NER research. By comparing the choices of DL architectures, we aim to identify factors affecting NER performance as well as issues.

1.1 Formulation of Named Entity

A named entity is a word or a phrase that clearly identifies one item from a set of other items that have similar attributes. Examples of named entities are organization, person, and location names in general domain, gene, protein, drug and disease names in biomedical domain. NER is the process of locating and classifying named entities in text into predefined entity categories. Formally, given a sequence of tokens $s = (w_1, w_2, \dots, w_n)$, NER is to output a list of tuples



2. Literature Survey

[1] “Few-Shot Named Entity Recognition: A Comprehensive Study”.

AUTHORS: J. Huang et al.

This paper presents a comprehensive study to efficiently build named entity recognition (NER) systems when a small number of in-domain labeled data is available. Based upon recent Transformer-based self-supervised pre-trained language models (PLMs), we investigate three orthogonal schemes to improve the model generalization ability for few-shot settings: (1) meta learning to construct prototypes for different entity types, (2) supervised pre-training on noisy web data to extract entity-related generic representations and (3) self-training to leverage unlabelled in-domain data. Different combinations of these schemes are also considered. We perform extensive empirical comparisons on 10 public NER datasets with various proportions of labeled data, suggesting useful insights for future research. Our experiments show that (i) in the few-shot learning setting, the proposed NER schemes significantly improve or outperform the commonly used baseline, a PLM-based linear classifier fine-tuned on domain labels; (ii) We create new state-of-the-art results on both few-shot and training-free settings compared with existing methods. We will release our code and pre-trained models for reproducible research.

[2] “A survey of named entity recognition and classification”.**AUTHORS: D. Nadeau and S. Sekine.**

In recent years, it has been seen that deep neural networks are lacking robustness and are likely to break in case of adversarial perturbations in input data. Strong adversarial attacks are proposed by various authors for computer vision and Natural Language Processing (NLP). As a counter-effort, several defense mechanisms are also proposed to save these networks from failing. In contrast with image data, generating adversarial attacks and defending these models is not easy in NLP because of the discrete nature of the text data. However, numerous methods for adversarial defense are proposed of late, for different NLP tasks such as text classification, named entity recognition, natural language inferencing, etc. These methods are not just used for defending neural networks from adversarial attacks, but also used as a regularization mechanism during training, saving the model from overfitting. The proposed survey is an attempt to review different methods proposed for adversarial defenses in NLP in the recent past by proposing a novel taxonomy. This survey also highlights the fragility of the advanced deep neural networks in NLP and the challenges in defending them.

[3] “Named entity recognition in query”.**AUTHORS: J. Guo, G. Xu, X. Cheng, and H. Li,**

There has been a growing academic interest in the recognition of nested named entities in many domains. We tackle the task with a novel local hypergraph-based method: We first propose start token candidates and generate corresponding queries with their surrounding context, then use a query-based sequence labelling module to form a local hypergraph for each candidate. An end token estimator is used to correct the hypergraphs and get the final predictions. Compared to span-based approaches, our method is free of the high computation cost of span sampling and the risk of losing long entities. Sequential prediction makes it easier to leverage information in word order inside nested structures, and richer representations are built with a local hypergraph. Experiments show that our proposed method outperforms all the previous hypergraph-based and sequence labelling approaches with large margins on all four nested datasets. It achieves a new state-of-the-art F1 score on the ACE 2004 dataset and competitive F1 scores with previous state-of-the-art methods on three other nested NER datasets: ACE 2005, GENIA, and KBP 2017.

3. Proposed System

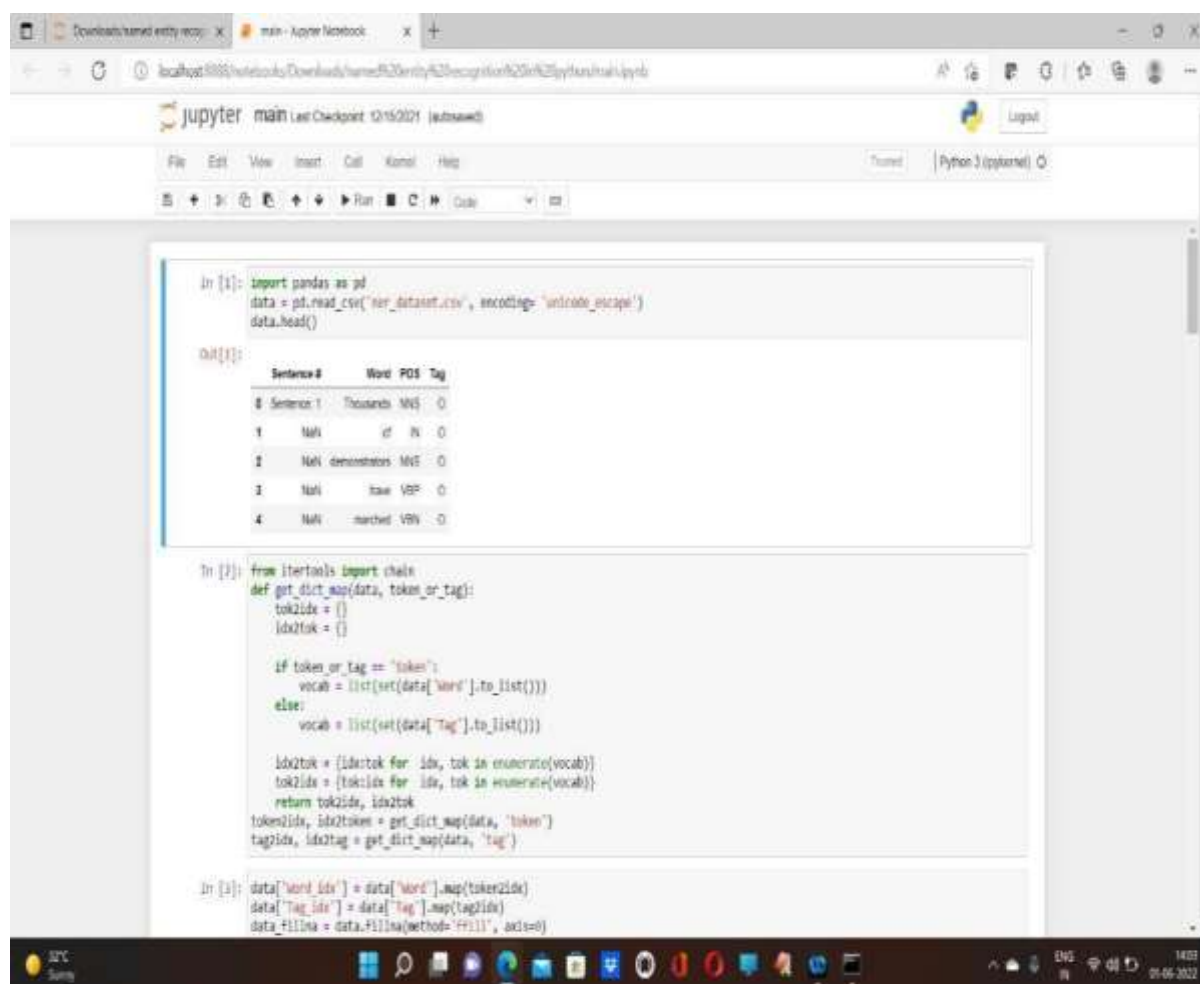
In this proposed system, deep learning (DL, also named deep neural network) has attracted significant attention due to its success in various domains. DL-based NER systems with minimal feature engineering have been flourishing. Over the past few years, a considerable number of studies have applied deep learning to NER and successively advanced the state-of-the-art performance. This trend motivates us to conduct a survey to report the current status of deep learning techniques in NER research. By comparing the choices of DL architectures, we aim to identify factors affecting NER performance as well as issues and challenges. We intensely review applications of deep learning techniques in NER, to enlighten and guide researchers and practitioners in this area. Specifically, we consolidate NER corpora, off-the-shelf NER systems (from both academia and industry) in a tabular form, to provide useful resources for NER research community. We then present a comprehensive survey on deep learning techniques for NER. To this end, we propose a new taxonomy, which systematically organizes DL-based NER approaches along three axes: distributed representations for input, context encoder (for capturing contextual dependencies for tag decoder), and tag decoder (for predicting labels of words in the given sequence). In addition, we also survey the most representative methods for recent applied deep learning techniques in new NER problem settings and applications.

Finally, we present readers with the challenges faced by NER systems and outline future directions in this area.

3.1 Advantages of Proposed System

NER acts as an important pre-processing step for a variety of downstream applications such as information retrieval, question answering, machine translation, etc. Here, we use semantic search as an example to illustrate the importance of NER in supporting various applications. Semantic search refers to a collection of techniques, which enable search engines to understand the concepts, meaning, and intent behind the queries from users. About 71% of search queries contain at least one named entity. Recognizing named entities in search queries would help us to better understand user intents, hence to provide better search results. To incorporate named entities in search, entity-based language models, which consider individual terms as well as term sequences that have been annotated as entities (both in documents and in queries). There are also studies utilizing named entities for an enhanced user experience, such as query recommendation, query autocompletion, and entity cards.

4. Results



```
In [1]: import pandas as pd
data = pd.read_csv('ner_dataset.csv', encoding='unicode_escape')
data.head()

Out[1]:
```

Sentence #	Word	POS	Tag	
0	Sentence:1	Thousands	NNS	O
1	Nali	if	IN	O
2	Nali	demonstrates	MVC	O
3	Nali	have	VP	O
4	Nali	matched	VBN	O

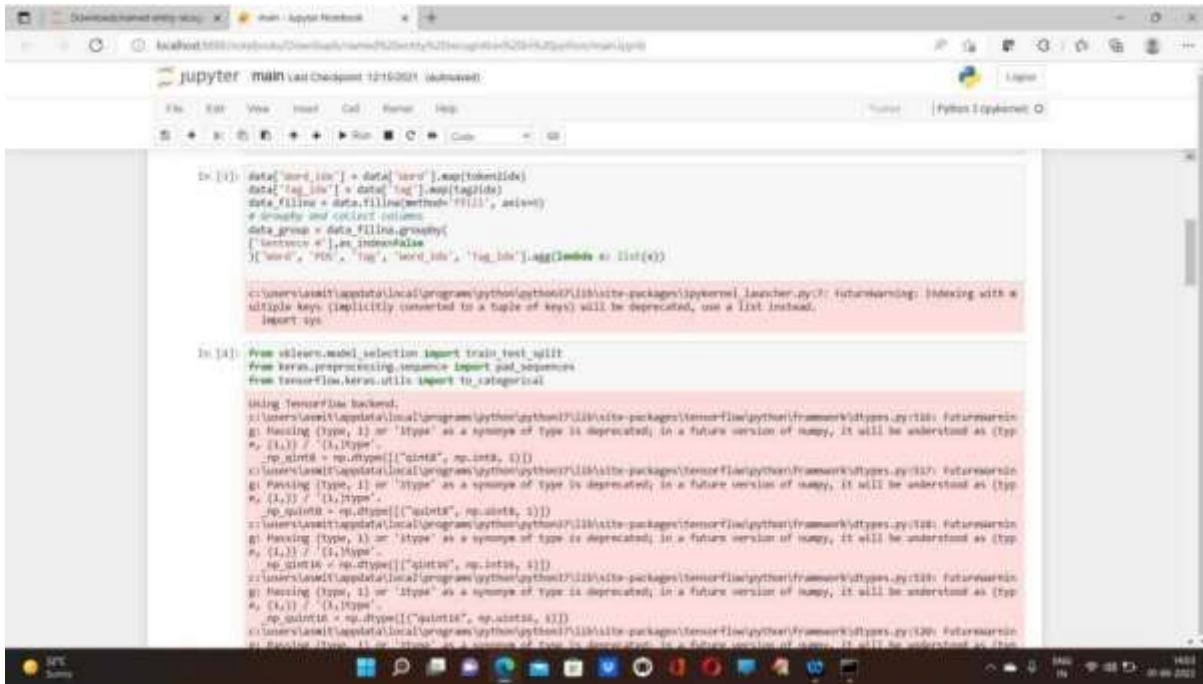
```
In [2]: from itertools import chain
def get_dict_map(data, token_or_tag):
    tok2idx = {}
    idx2tok = {}

    if token_or_tag == 'token':
        vocab = list(set(data['word']).to_list())
    else:
        vocab = list(set(data['Tag']).to_list())

    idx2tok = {idx:tok for idx, tok in enumerate(vocab)}
    tok2idx = {tok:idx for idx, tok in enumerate(vocab)}
    return tok2idx, idx2tok

tokens2idx, idx2token = get_dict_map(data, 'token')
tags2idx, idx2tag = get_dict_map(data, 'Tag')

In [3]: data['word_idx'] = data['word'].map(tokens2idx)
data['Tag_idx'] = data['Tag'].map(tags2idx)
data.fillna(method='ffill', axis=0)
```

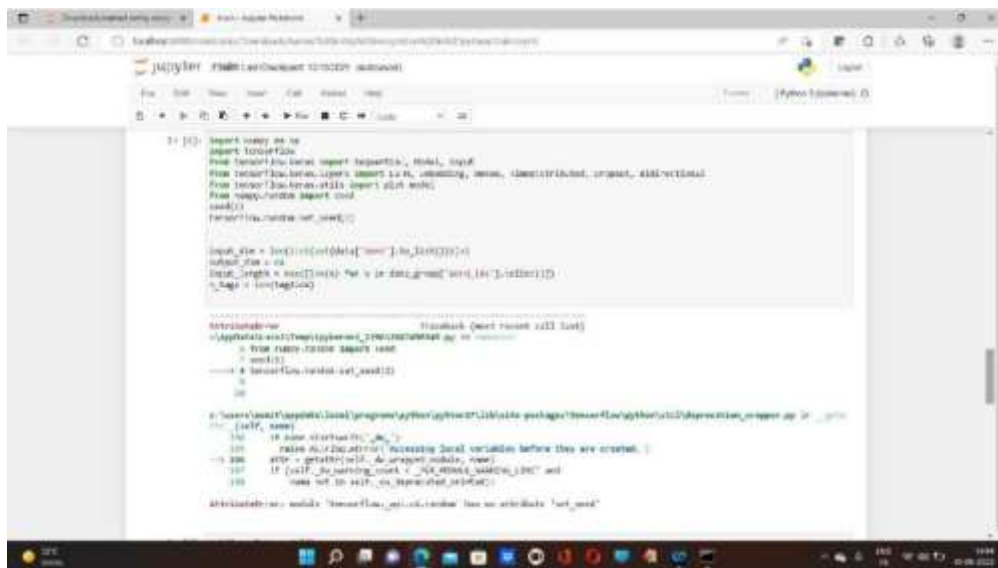


```
In [1]: data['word_id'] = data['word'].apply(token2id)
data['tag_id'] = data['tag'].apply(tag2id)
data['filler'] = data.fillna(method='ffill', axis=0)
# identify and collect columns
data_group = data.fillna(method='ffill', axis=0)
[['tokens', 'POS', 'tag', 'word_id', 'tag_id'].agg(lambda x: list(x))

C:\Users\asaf\AppData\Local\Programs\Python\Python311\site-packages\ipykernel_launcher.py:7: FutureWarning: indexing with a
multiple keys (explicitly converted to a tuple of keys) will be deprecated, use a list instead.
import sys

In [4]: from sklearn.model_selection import train_test_split
from keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.utils import to_categorical

Using TensorFlow backend.
c:\users\asaf\appdata\local\program\python\python311\site-packages\tensorflow\python\framework\dtypes.py:118: FutureWarning
g: Passing (Type, 1) or 'dtype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (Typ
e, (1,)) / (1, dtype).
..._np_dtype = np.dtype(['tokens', np.int32, 0])
c:\users\asaf\appdata\local\program\python\python311\site-packages\tensorflow\python\framework\dtypes.py:117: FutureWarning
g: Passing (Type, 1) or 'dtype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (Typ
e, (1,)) / (1, dtype).
..._np_dtype = np.dtype(['tokens', np.int32, 0])
c:\users\asaf\appdata\local\program\python\python311\site-packages\tensorflow\python\framework\dtypes.py:118: FutureWarning
g: Passing (Type, 1) or 'dtype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (Typ
e, (1,)) / (1, dtype).
..._np_dtype = np.dtype(['tokens', np.int32, 0])
c:\users\asaf\appdata\local\program\python\python311\site-packages\tensorflow\python\framework\dtypes.py:117: FutureWarning
g: Passing (Type, 1) or 'dtype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (Typ
e, (1,)) / (1, dtype).
```



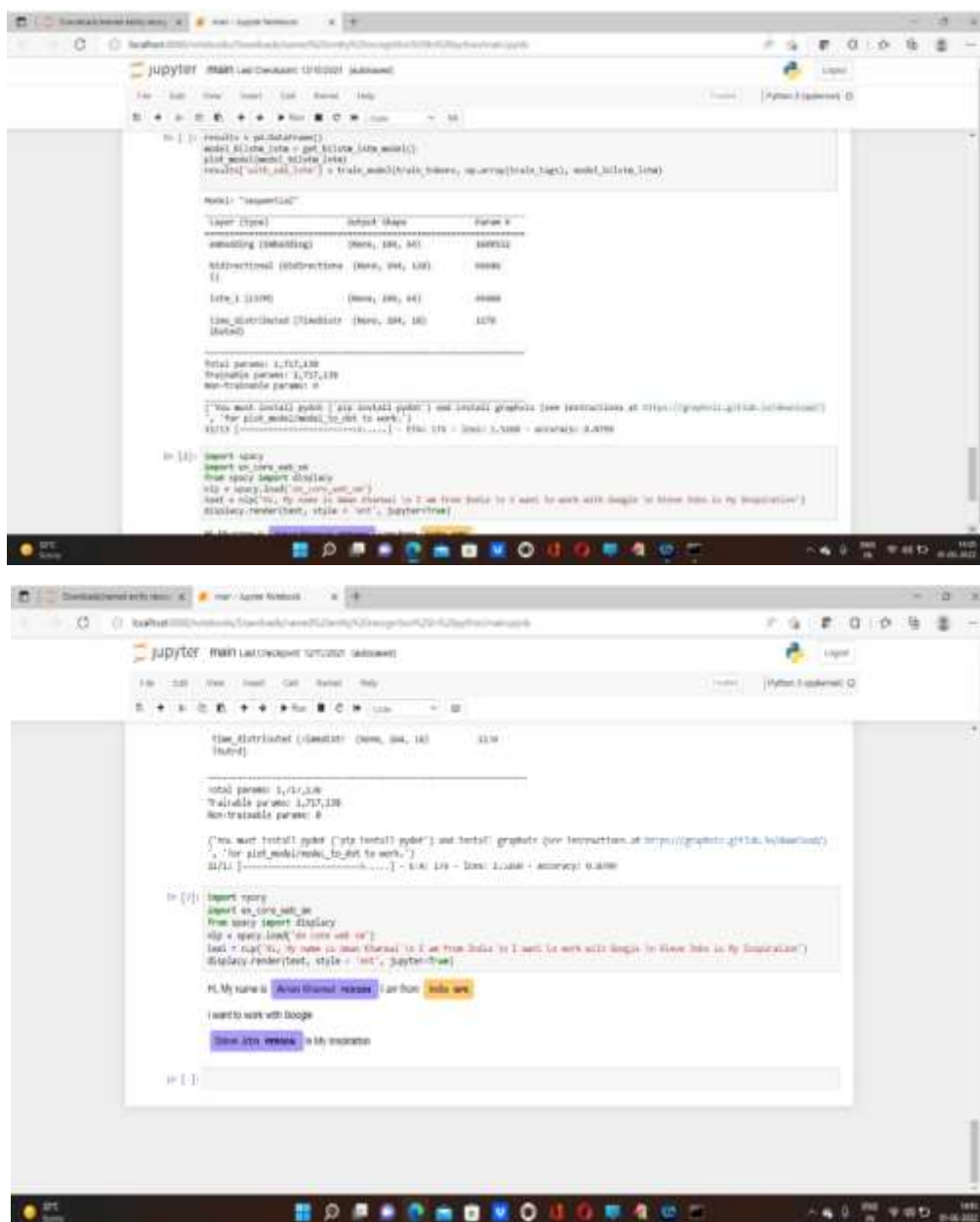
```
In [1]: import copy as cp
import tensorflow
from tensorflow.keras import Sequential, Model, Input
from tensorflow.keras.layers import LSTM, SimpleRNN, GRU, LSTM, Bidirectional, Input, Embedding, Dense, Dropout, BatchNormalization, SpatialDropout1D
from keras.callbacks import ModelCheckpoint
from tensorflow.keras.optimizers import Adam

vocab_size = len(set(tokens['word']).union(tokens['tag']))
vocab_size += 1
embed_dim = 128
input_shape = (vocab_size, None) # or data_group['tokens'].infer()
output_shape = (vocab_size, None)

# compile the model
model = Sequential([
    Input(shape=input_shape),
    Embedding(vocab_size, embed_dim),
    LSTM(embed_dim, return_sequences=True),
    Dense(vocab_size),
])

# compile the model
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])

# train the model
model.fit(data_group['tokens'].infer(), data_group['tags'].infer(), epochs=100, validation_data=(data_group['tokens'].infer(), data_group['tags'].infer()), callbacks=[ModelCheckpoint('model.h5', save_best_only=True)], verbose=1)
```



5. Conclusion and Future Scope

Our project aims to review recent studies on deep learning-based NER solutions to help new researchers building a comprehensive understanding of this field. We include in this project the background of the NER research, a brief of traditional approaches, current state-of-the-arts, and challenges and future research directions. First, we consolidate available NER resources, including tagged NER corpora and off-the-shelf NER systems, with focus on NER in general domain and NER in English. We present these resources in a tabular form and provide links to them for easy access. Second, we introduce preliminaries such as definition of NER task, evaluation metrics, traditional approaches to NER, and basic concepts in deep learning. Third, we review the literature based on varying models of deep learning and map these studies according to a new taxonomy. We further survey the most representative methods for recent applied deep learning techniques in new problem settings and applications. Finally, we summarize the applications of NER and present readers with

challenges in NER and future directions. We hope that this project can provide a good reference when designing DL-based NER models.

References

- [1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” arXiv:1810.04805 [cs], Oct. 2018, Accessed: Mar. 04, 2021. [Online]. Available: <http://arxiv.org/abs/1810.04805>.
- [2] Beltagy, K. Lo, and A. Cohan, “SciBERT: A Pretrained Language Model for Scientific Text,” arXiv:1903.10676 [cs], Sep. 2019, Accessed: Mar. 04, 2021. [Online]. Available: <http://arxiv.org/abs/1903.10676>.
- [3] R.J. Bayardo, Y. Ma, and R. Srikant, “Scaling up all pairssimilarity search,” in Proceedings of the 16th international conference on World Wide Web — WWW ’07, Banff, Alberta, Canada, 2007, p. 131, doi: 10.1145/1242572.1242591.
- [4] G. Hripesak and A. S. Rothschild, “Agreement, the F-Measure, and Reliability in Information Retrieval,” *Journal of the American Medical Informatics Association*, vol. 12, no. 3, pp. 296–298, May 2005, doi: 10.1197/jamia.M1733.
- [5] J. Huang et al., “Few-Shot Named Entity Recognition: A Comprehensive Study,” arXiv:2012.14978 [cs], Dec. 2020, Accessed: Mar. 04, 2021. [Online]. Available: <http://arxiv.org/abs/2012.14978>.
- [6] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, “Span BERT: Improving Pre-training by Representing and Predicting Spans,” *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 64–77, Dec. 2020, doi: 10.1162/tacl_a_00300