Proceedings of the Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI 2024)
IEEE Xplore Part Number : CFP24VG0-ART ; ISBN : 979-8-3315-4066-1

2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI) | 979-8-3315-4066-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICOICI62503.2024.10696673

Text Summarization based Named Entity Recognition for Certain Application using BERT

Indira Priyadarshini Tummala
CSE-AI&ML Department
Anurag Engineering College
Kodad, India
priya230@gmail.com

***Abstract*—Text summarization is gaining attention from readers who want to understand the main concept of the entire text. One of the natural language processing forms, "Named Entity Recognition" (NER) is used to summarize the text with deep learning. Semantic and syntactic relationships between the words are well understood by NER, and it uses data pre-processing, which mainly detects different entities from the text and classifies them into various categories. In this work, text reading, text processing, text formulation, and text evaluation are performed in the summarization of the text using BERT, Transformers algorithms. Deep learning concepts are incorporated to reduce the training loss and increase validation accuracy with fine tuning of BERT algorithm. NER is implemented on Spacy and NLTK frameworks, and the code was executed on Colab/Python)**

***Keywords*—Named Entity Recognition, BERT, Transformers, Natural Language Processing**

I. INTRODUCTION

A named entity is a word or phrase that precisely distinguishes one object from a collection of like objects. Examples of general named entities include names of organizations, persons, and locations; examples of named entities in the biomedical domain are names of genes, proteins, medications, and illnesses. NER is the technique of identifying and classifying named entities in text into specified entity categories. One of the areas with the aim is Named Entity Recognition, which seeks semantics of interest in unstructured texts; it is the basis for many other important fields of information management including semantic annotations, question answering, ontology population, and opinion mining.

Named Entity Recognition aims to detect specific terms in text that fall into predetermined semantic categories, such as person, place, organization, and others [1,2]. NER serves as both a standalone tool for extracting information and as a component in several natural language processing applications, including text comprehension [3], information retrieval, automated text summarization, question answering, machine translation, and knowledge base development. Named Entity Recognition (NER) is a crucial first stage in several subsequent tasks including information retrieval, question answering, machine translation, and others [4]. By examining NER definitions, it may classify them based on four specific criteria: grammatical category, rigid designation, unique identification, and realm of application.

The global natural language processing (NLP) market is expected to be worth $43 billion by 2025, 14 times what it was in 2017 [5, 6]. By enabling machines to understand and process human languages, NLP has a wide range of applications in every industry, from chatbots to document processing.

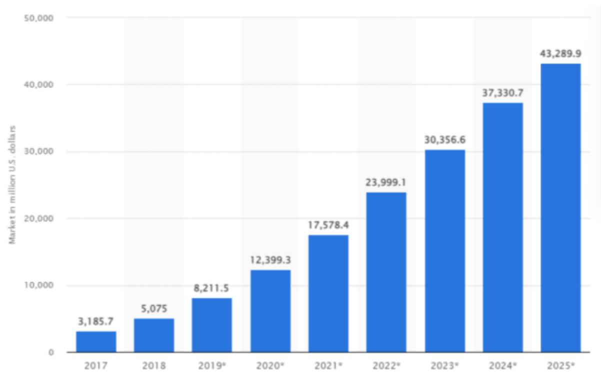

Fig.1: Expected NLP Global Revenue Market [3]

Named Entity Recognition (NER) is a process that involves scanning a whole text and identifying certain entities, such as names of people, places, organizations, and other relevant information [7]. The Named Entity Recognition (NER) algorithm identifies the borders of sentences in a given text by analyzing the rules of capitalization. Determining the sentence boundaries will aid in the NER process of locating and extracting pertinent information from the material for subsequent stages. Named Entity Recognition (NER) classifies entities into predetermined categories. To label words or phrases, it is necessary to provide precise definitions for entity categories such as place, people, event, time, organization, and others [8, 9]. The entity extraction model may be trained using specified categories to identify entities, such as individuals, locations, and organizations, in unprocessed texts. BERT is a language model of significant size that utilizes bidirectional encoder representations from transformers. This pre-trained model may be customized and used for many tasks, such as sentiment analysis, question answering, word classification, and other similar applications [10]. BERT is the preeminent technique for transfer learning in the field of Natural Language Processing (NLP).

This work focuses on making the NER technique simple and reliable. It may say that employing a transformer for NER can be reliable because of the success of transformers in machine learning. BERT is also one of the most successful transformers for completing various NLP jobs to produce best-in-class results [11]. In this work, transformer library is also included, which will allow us to use several transformers for pre-processing the data.

II. Lieterature Review

Abstractive text summarization has garnered significant interest in recent years with the development of the seq2seq paradigm. Several neural network-based models have shown superior performance compared to traditional approaches. Recently, there has been a use of various encoder-decoder designs, such as convs2s and transformers. The given text is a list containing the elements 3 and 4. Deep learning, commonly referred to as deep neural network (DL), has garnered significant interest in recent years because to its remarkable achievements across several domains. Deep learning-based named entity recognition (NER) systems that need little feature engineering have proven very successful since the work of Collobert et al [12]. In recent years, several studies have used deep learning techniques for Named Entity Recognition (NER), resulting in a consistent improvement in the state-of-the-art performance. Due to this inclination, a survey to evaluate the current status of deep learning methods in Named Entity Recognition (NER) is initiated. However, despite the extensive duration of NER research, there have been little assessments conducted in this field to our understanding. In 2007, Nadeau and Sekine [13, 14] created what is widely considered to be the most famous one. This study offers a comprehensive analysis of the transition from manual rule creation to the use of machine learning algorithms. In 2013, Marrero et al. [15] provided a concise overview of the progress, challenges, and potential future developments in the field of Named Entity Recognition (NER). In 2015, a concise review was published by Patawar and Potey [16].

Two recent concise studies focus on new domains and the mention of complicated entities. To summarize, current studies mostly concentrate on machine learning models that are based on features, rather than the more recent named entity recognition (NER) systems that use deep learning (DL) techniques. The two latest polls conducted in 2018 have more relevance to this work. Goyal et al. conducted a comprehensive analysis of the advancements made in the field of Named Entity Recognition (NER). However, they omitted the incorporation of recent breakthroughs in deep learning algorithms. Yadav and Bethard provided a concise summary of recent advancements in Named Entity Recognition (NER) that are based on word representations in sentences [17]. The study does not include an analysis of context encoders and tag decoders since it focuses only on distributed representations for input, such as character- and word-level embeddings. While transformers are successful in machine translation, they do not do well in abstractive text summarization due to their limited capability in modeling the sequential context at the word level. Deep learning-based Named Entity Recognition (NER) models have grown more common and have achieved state-of-the-art results. Deep learning surpasses feature-based approaches in its ability to automatically uncover latent characteristics. Subsequently, the definition and significance of deep learning in the context of Named Entity Recognition (NER) will be discussed. The BERT method is used in this context to pre-train the model, fine-tune it, and apply it to various tasks such as sentiment analysis, question answering, and word classification..

III. Implementation

Python-based NER has also gotten a lot of attention recently. Predefined categories such as persons, organisations, places, time notations, values, and others are used in named entity recognition. Named entities are nothing more than words used in a phrase to categorise them. A simple transformer library and BERT to create NER will be utilized, which is also known as token categorization. The BERT algorithm will be fine-tuned based on the training loss and validation results. Dataset after installing simple transformers and their dependencies need to be imported. A total of 48000 sentences are analysed in the datasets. These Kaggle data sets have been harmonised. Start looking at the sentence type, such as whether it's written in the past or future tense. Colab is used to programme, and the resulting results are displayed to check the runtime and outputs of various algorithms. The architecture of the named entity recognition with BERT is presented in Fig. 2, and the implementation of NER with BERT is presented in Fig. 3.

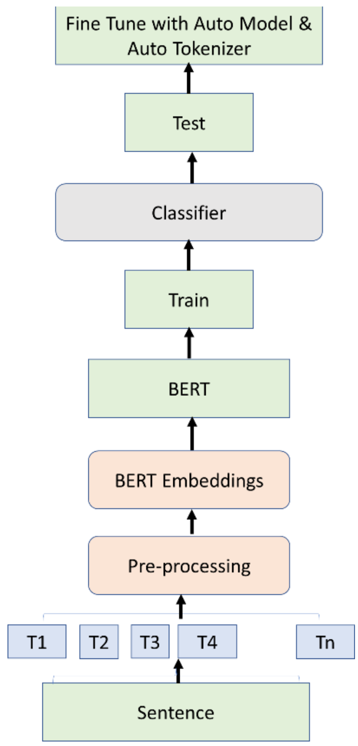

Fig.2: Architecture of the NER with BERT

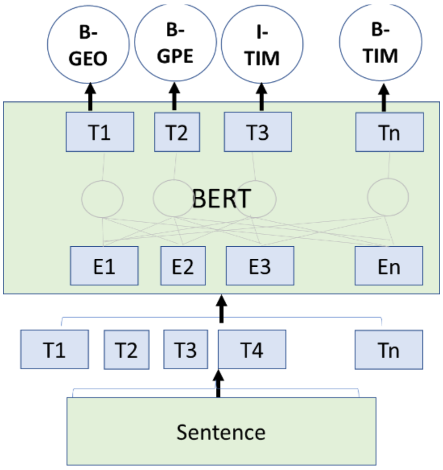

Fig.3: Implementation of NER with BERT

Import the dataset to be used for NER using the pandas package. The dataset should be read from the local disc. It is also required to figure out what format is needed for NER systems to be implemented and predicted. The Tag & PoS column from the data read will be our objective here. The data format must first be understood. Apart from the initial word, all other words in sentence# are referred to as NaN, according to the data read. Replace NaN with the appropriate sentences. B-geo, which previously indicated geographical position, is used to denote produced tags for places. Figure 1 displays the read dataset. Because some of the words do not contain sentence#, this dataset will need to be pre-processed. Wherever NaN valves exist in pre-processing, relevant sentence numbers must be placed with filling. data.fillna will be utilized for this.

IV. Results

For the dataset, BERT and Transformer algorithm are implemented and the results are presented. While computing the BERT algorithm it is observed the validation accuracy is 0.92319, which is low and requires further fine tuning. For fine tuning of the BERT algorithm AUTOTOKINZER, PIPELINE and AUTO MODEL FOR TOKEN CLASSIFICATION are considered so that the validation accuracy can be enhanced. In this analysis, the validation accuracy has 0.99976, which gives more accuracy in the summarization. Apart from the data set, the same is implemented on resumes and the found better summarization after fine tuning the BERT algorithm. When dataset is applied for NER, the labels have been identified as shown in Fig. 4

```
label = data["Labels"].unique().tolist()
label
```

```
['O',
 'B-GEO',
 'B-GPE',
 'B-PER',
 'I-GEO',
 'B-ORG',
 'I-ORG',
 'B-TIM',
 'B-ART',
 'I-ART',
 'I-PER',
 'I-GPE',
 'I-TIM',
 'B-NAT',
 'B-EVE',
 'I-EVE',
 'I-NAT']
```

Fig.4: NER Labels classification from dataset

The abbreviations of Fig. 4 are listed below for easy understanding.

Geo : Geographical Entity
Org : Organization
Per : Person
Gpe : Geopolitical Entity
Tim : Time indicator
Art : Artifact
Eve : Event
Nat : Natural Phenomenon
I : Tag is inside a chunk.
B : Tag is the beginning of a chunk.
O : Token belongs to no chunk (outside)

Once entities are recognized train the data and observe the accuracy. The accuracy for 5 epochs is done and the validation accuracy & train loss are presented in Fig. 5. Validation accuracy is about 0.923 and the training loss is reduced as the epochs increases.

```
Epoch:   0%|          | 0/5 [00:00<?, ?it/s]
Train loss: 0.8511637701438024

Epoch:  20%|█         | 1/5 [00:08<00:34,  8.66s/it]
Validation loss: 0.5098541527986526
Validation Accuracy: 0.9231944444444444
F1-Score: 0.9242331288343558
Train loss: 0.5241081989728488

Epoch:  40%|██        | 2/5 [00:16<00:25,  8.43s/it]
Validation loss: 0.4943576753139496
Validation Accuracy: 0.9231944444444444
F1-Score: 0.9242331288343558
Train loss: 0.5169478012965276

Epoch:  60%|███       | 3/5 [00:24<00:16,  8.26s/it]
Validation loss: 0.51574215292930б
Validation Accuracy: 0.9231944444444444
F1-Score: 0.9242331288343558
Train loss: 0.5097572069901687

Epoch:  80%|████      | 4/5 [00:32<00:08,  8.15s/it]
Validation loss: 0.48067526519298553
Validation Accuracy: 0.9231944444444444
F1-Score: 0.9242331288343558
Train loss: 0.484199753174415
```

```
Epoch: 100%|██████████| 5/5 [00:40<00:00,  8.06s/it]
Validation loss: 0.5480965077877045
Validation Accuracy: 0.9231944444444444
F1-Score: 0.9242331288343558
```

Fig.5. Validation accuracy & Training loss of dataset without fine tuning BERT for different epochs

From figure 5 it is observed that the accuracy of the validation is little bit low i.e. 0.923, which requires enhancement. For this the BERT algorithm is fine tuned to enhance the accuracy score. Fine tuning is done with the help of Auto Tokenizer, Pipes and Auto model for token classification. Fine-tuned accuracy is found to be 0.99976 which is shown in figure 6.

```
Epoch:   0%|          | 0/5 [00:00<?, ?it/s]
Train loss: 0.7356117704138937
Epoch:  20%|██        | 1/5 [00:08<00:34,  8.66s/it]
Validation loss: 0.5098541527986526
Validation Accuracy: 0.9997663333333333
F1-Score: 0.9990788124283485
Train loss: 0.464199753174415
Epoch:  40%|████      | 2/5 [00:16<00:25,  8.43s/it]
Validation loss: 0.4943576753139496
Validation Accuracy: 0.9997663333333333
F1-Score: 0.9990788124283485
Train loss: 0.441199743411551
Epoch:  60%|██████    | 3/5 [00:24<00:16,  8.26s/it]
Validation loss: 0.515742152929306
Validation Accuracy: 0.9997663333333333
F1-Score: 0.9990788124283485
Train loss: 0.428199744513517
Epoch:  80%|████████  | 4/5 [00:32<00:08,  8.15s/it]
Validation loss: 0.48067526519298553
Validation Accuracy: 0.9997663333333333
F1-Score: 0.9990788124283485
Train loss: 0.419975432131475
Epoch: 100%|██████████| 5/5 [00:40<00:00,  8.06s/it]
Validation loss: 0.5480965077877045
Validation Accuracy: 0.9997663333333333
F1-Score: 0.9990788124283485
```

Fig.6. Validation accuracy & Training loss of dataset with fine tuning BERT for different epochs

The same analysis is implemented on Resume applications, so that the summarization of the resume along with entities are predicted. For this analysis, two different domain resumes are fed and found the summarization with and without fine tuning which is shown in Fig. 7 -10.

```
Govardhana K Senior Software Engineer Bengaluru, Karnataka, Karnataka
- Email me on Indeed: indeed.com/r/Govardhana-K/ b2de315d95905b68
Total IT experience 5 Years 6 Months Cloud Lending Solutions INC 4
Month • Salesforce Developer Oracle 5 Years 2 Month • Core Java
Developer Languages Core Java, Go Lang Oracle PL-SQL programming,
Sales Force Developer with APEX. Less than 1 year), Data Structures
(3 years), FLEXCUBE (5 years), Oracle (5 years), Algorithms (3 years)
LINKS https://www.linkedin.com/in/govardhana-k-61024944/ ADDITIONAL
INFORMATION Technical Proficiency: Languages: Core Java, Go Lang,
Data Structures & Algorithms, Oracle PL-SQL programming, Sales Force
with APEX.
```

Fig. 7. BERT Summarization for Resume 1 without fine tuning.

```
Govardhana K Senior Software Engineer Bengaluru, Karnataka, Karnataka
- Email me on Indeed: indeed.com/r/Govardhana-K/ b2de315d95905b68
Total IT experience 5 Years 6 Months Cloud Lending Solutions INC 4
Month • Salesforce Developer Oracle 5 Years 2 Month • Core Java
Developer Languages Core Java, Go Lang Oracle PL-SQL programming,
Sales Force Developer with APEX. Less than 1 year), Data Structures
(3 years), FLEXCUBE (5 years), Oracle (5 years), Algorithms (3 years)
LINKS https://www.linkedin.com/in/govardhana-k-61024944/ Technical
Proficiency in Core Java, Go Lang, Data Structures Languages &
Algorithms, Oracle PL-SQL programming, Sales Force with APEX.
```

Fig. 8. BERT Summarization for Resume 1 with fine tuning.

Alice Clark

AI / Machine Learning

Delhi, India Email me on Indeed

- 20+ years of experience in data handling, design, and development
- Data Warehouse: Data analysis, star/snow flake scema data modelling and design specific to data warehousing and business intelligence
- Database: Experience in database designing, scalability, back-up and recovery, writing and optimizing SQL code and Stored Procedures, creating functions, views, triggers and indexes. Microsoft Rewards members can earn points when searching with Bing, browsing with Microsoft Edge and making purchases at the Xbox Store, the Windows Store and the Microsoft Store.

Fig. 9. BERT Summarization for Resume 2 without fine tuning.

Alice Clark

AI / Machine Learning

Delhi, India Email me on Indeed

- 20+ years of experience in data handling, design, and development
- Data Warehouse: Data analysis, star/snow flake scema data modelling and design specific to data warehousing and business intelligence
- Database: Experience in database designing, scalability, back-up and recovery, writing and optimizing SQL code and Stored Procedures, creating functions, views, triggers and indexes. Microsoft Rewards Live dashboards:

Description: - Microsoft rewards is loyalty program that rewards Users for browsing and shopping online.

Fig. 10. BERT Summarization for Resume 2 with fine tuning.

V. Conclusion

Named entity recognition with BERT and Transformers algorithms are implemented to datasets and its runtime and output text summarization are presented. Validation accuracy has been improved from 0.92319 to 0.99976 for the dataset considered. This validation accuracy increase is due to fine tuning the existing BERT algorithm by using auto tokenizer, pipeline and auto model for token classification. Bert based NER is used by considering the pretrained data of NER. Fine-tuned BERT algorithm results on the dataset considered found to be satisfactory, as it gave the entities, accuracy score, index, start & end points of the word.

References

[1] Allahyari M, Pouriyeh S, Assefi M, Safaei S, Trippe ED, Gutierrez JB, Kochut K (2017) "Text summarization techniques: A brief survey," arXiv preprint arXiv:1707.02268

[2] Akhil K, Rajimol R, Anoop V (2020) Parts-of-speech tagging for malayalam using deep learning techniques. International Journal of Information Technology 12(3):741–748

[3] Kumar A, Kumar R (2021) A novel approach for isl alphabet recognition using extreme learning machine. International Journal of Information Technology 13(1):349–357

[4] Vilca GCV, Cabezudo MAS (2017) "A study of abstractive summarization using semantic representations and discourse level information," In International Conference on Text, Speech, and Dialogue, Springer, pp. 482-490.

[5] Chopra S, Auli M, Rush AM (2016) "Abstractive sentence summarization with attentive recurrent neural networks," In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 93-98

[6] Meshram S, Kumar MA (2021) "Long short-term memory network for learning sentences similarity using deep contextual embeddings," International Journal of Information Technology, pp. 1-9

[7] Sintayehu H, Lehal G (2020) "Named entity recognition: A semi-supervised learning approach," International Journal of Information Technology, pp. 1-7

[8] Anand D, Wagh R (2019) "Effective deep learning approaches for summarization of legal texts," Journal of King Saud University-Computer and Information Sciences

[9] Alami N, Meknassi M, En-nahnahi N (2019) Enhancing unsupervised neural networks based text summarization with word embedding and ensemble learning. Expert systems with appli- cations 123:195–211

[10] Gambhir M, Gupta V (2017) Recent automatic text summarization techniques: A survey. Artificial Intelligence Review 47(1):1–66

[11] El-Kassas WS, Salama CR, Rafea AA, Mohamed HK (2020) "Automatic text summarization: A comprehensive survey," Expert Systems with Applications, p. 113 679

[12] Song S, Huang H, Ruan T (2019) Abstractive text summarization using lstm-cnn based deep learning. Multimedia Tools and Applications 78(1):857–875

[13] Yousefi-Azar M, Hamey L (2017) Text summarization using unsupervised deep learning. Expert Systems with Applications 68:93–105

[14] Cai T, Shen M, Peng H, Jiang L, Dai Q (2019) "Improving transformer with sequential context representations for abstractive text summarization," In NLPCC.

[15] Cao Y, Liu H, Wan X (2020) "Jointly learning to align and summarize for neural crosslingual summarization," In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 6220-6231.

[16] Patel D, Chhinkaniwala H (2018) Fuzzy logic-based single document summarisation with improved sentence scoring technique. International Journal of Knowledge Engineering and Data Mining 5(1–2):125–0138.

[17] Joshi A, Fidalgo E, Alegre E, Fernández-Robles L (2022) Summcoder: An unsupervised framework for extractive text summarization based on deep auto-encoders. Expert Systems with Applications 129:200–215.