Automated Relational Triple Extraction from Unstructured Text Using Transformer

Akshay Hari(✉) and Priyanka Kumar

Department of Computer Science and Engineering,
Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India
cb.en.p2aid20009@cb.students.amrita.edu,
k_priyanka@cb.amrita.edu

Abstract. In modern times, large amount of textual data is generated. Quick comprehension of knowledge from this massive amount of data is difficult for human beings as well as machines. In this paper, we propose a deep learning based framework for joint extraction of entities and relations from unstructured text. This will be implemented with state-of-the-art Transformer based language model. Our model is a light version of the existing state-of-the-art models for the same task with only half of their trainable parameter while maintaining good evaluation scores. The model is trained and tested on NYT and WebNLG dataset and evaluation is done using metrics such as Precision, Recall and F1 scores.

Keywords: Deep learning · NLP · Relation extraction · Transformers

1 Introduction

In the modern age, there has been an increase in data. These data are mostly stored in the electronic form. The most common is the textual form in which information is stored in an unstructured manner. In order to this data to be useful, we should be able to retrieve most important information from this text seamlessly.

In this work, we propose a deep-learning based approach to extract relational triples in text. Relational triples are entities in a sentence which are in the form subject - predicate - objects. These relational triples can then be used for knowledge engineering applications.

The relational triples are extracted from unstructured text using a DistilBERT [1, 2] based transformer language model. One of the major highlights of our transformer-based model is that it will be able to capture dependencies in long sentences. Our model is also capable of extracting sentences with overlapping entities. This is a case where triples share same entities and relations. This scenario is explained in detail in the coming section. The final important aspect of our model is the joint entity-relation extraction. In the earlier models, entities and relations were separately learned in a pipe-lined manner, which resulted in error propagation from one stage to another.

S. Mekhilef et al. (Eds.): ICEEE 2022, LNEE 893, pp. 472–480, 2022.
https://doi.org/10.1007/978-981-19-1742-4_40

1.1 Overlapping Entities

Earlier models for this task were not able to handle sentences with overlapping entities. This is a scenario where entities are shared by multiple triples in same sentence. This can be categorized into mainly two types: Single Entity Overlapping (SEO) and Entity Pair Overlapping (EPO). Single Entity Overlapping occurs when multiple triples have same entity shared as subject or object. Entity Pair Overlap occurs when multiple triples have same entity pairs. A visual representation of these scenarios is given in Fig. 1.

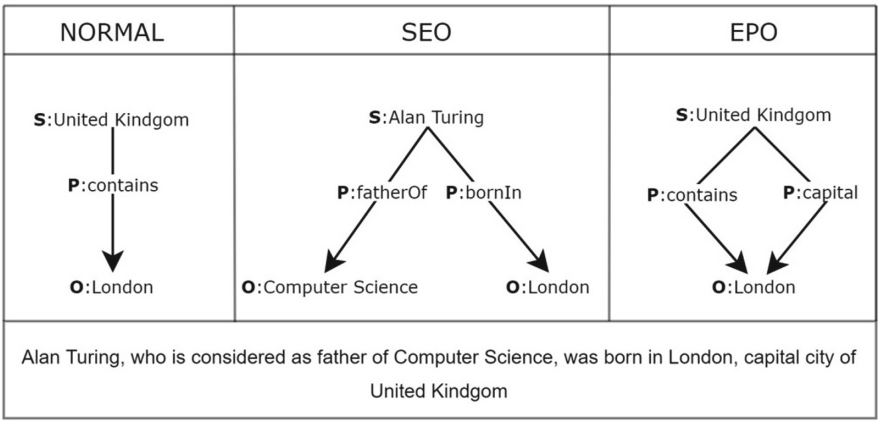

Fig. 1. Categories of triples in a sentence. Subject, Predicate and Object suffixes are added to the entities

1.2 Transformers

Most of the state-of-the-art models in the Natural Language Processing domain currently uses Transformer based language models. Transformer models are a new type of deep learning model [3] which uses attention mechanism to find global dependencies in input and output. The transformer model processes sentences in a non-sequential manner which helps in processing sentence as whole rather than word by word. All the above features were not prevalent in earlier deep neural network-based models for same task. However, most of the state-of the-art transformers have very high number of layers and parameters.

In our work, we are mainly using encoder mechanism of the transformer for language modelling. Our model can be summarized as follows. First raw textual inputs are converted into tokens using tokenizer. Then these tokens along with masks are fed into the encoder module to get the embedding. Since we're using DistilBERT based model, the output embedding is contextual in nature. This embedding along with the embedding of triplet labels are fed into the model for training. The loss is calculated in propagation with subject's and object's head and tail position in the sentence. The final output is the subject's and object's head and tail position in the sentence along with the relation.

2 Related Works

In the Information Extraction or Relation Extraction domain, one of the earlier notable work is [4] extracting features using Support Vector Machines. Later [5] approached the problem with a two-step solution, first is finding all entities using Named Entity Recognition (NER) and then classifying all the extracted entity pairs using relation classification (RC). These pipeline-based approaches however suffered from error propagation problem. To address this issue, joint models [6] have been proposed which learns entities and relations together. The earlier works, however did not addressed the problem of overlapping entities encountered in a sentence i.e. multiple triples in same sentence sharing same entities. This problem was only recently addressed using deep neural network based models in the works of [7], which is based on sequence-to-sequence learning with copy mechanism using Bi-directional LSTM. Later the evaluation scores were improved by [8] using Graph Convolutional Networks and Bi-LSTMs. The recent works by [9] and [10] further improves the evaluation scores using BERT based transformer language model. Other recent works involving the usage of transformers in knowledge extractions include [11–13].

3 Dataset

For training and testing of our relation extraction framework, we are using two public dataset, New York Times dataset and WebNLG dataset. The original NYT dataset [14] was created with distant supervision approach and WebNLG dataset [15] for Natural Language Generation. These datasets have been modified as per the requirement [7]. The resulting NYT dataset consists of 24 classes, 56195 training data, 5000 validation data and 5000 test data. The WebNLG dataset consists of total 5019 training data, 500 validation data and 703 test data. Detailed information is given in Table 1.

Total train data in each dataset used for training, however testing is done on individual component. The testing data can be classified into three types Normal, Entity Pair Overlap and Single Entity Overlap. The testing data can be further classified on basis of number of relational triples exists on a single sentence. All the testing data without categorized is marked as main. Tabulated information of the dataset is given in Table 2. In the Table 2, for the rows 'Triple-i', i denotes number of triples in a single sentence.

Table 1. Dataset information

	NYT		WebNLG	
Dataset	Train	Test	Train	Test
Normal	37013	3266	1596	246
Triple-SEO	9782	978	227	26
Triple-EPO	14735	1297	3406	457
Total	56195	5000	5019	703

Table 2. Categorization of testing data

Types	NYT	WebNLG
Main	5000	703
Triple-1	3244	266
Triple-2	1045	171
Triple-3	312	131
Triple-4	291	90
Triple-5	108	45
Normal	3266	246
SEO	1297	457
EPO	978	26

4 Model Architecture

For the relational extraction model, we followed the work of [9, 10] which was implemented using BERT based encoder and Graph Neural network. We have optimized the size of the same using DistilBERT based transformer framework without using Graph Network Layer from baseline as accuracy gains from Graph Neural Network was negligible in our experiments when considering number of trainable parameters it added. This allowed us to significantly reduce the trainable parameters without compromising much of accuracy as well as lowering the model training time.

For relation extraction framework (Fig. 2), our work consists of two parts: encoding words from input sentence into vector embeddings and encoding each relation into vectors and then subject and object tagger based relational triple extraction.

The problem can be formulated as mentioned. Given a sentence x, and set of all triplets (s,r,o) in training set T, our goal is to maximize the data-likelihood in the training set. This can be mathematically defined as mention in Eq. 1:

$$\begin{aligned} &\prod_{(s,r,o)\in T} p((s,r,o)|x) \\ &= \prod_{s\in T} p(s|x) \prod_{(r,o)\in T|s} p((r,o)|x,s) \\ &= \prod_{s\in T} p(s|x) \prod_{r\in T|s} p(o|x,s,r) \prod_{r\in R\setminus T|s} p(o_{\emptyset}|x,s,r) \end{aligned} \tag{1}$$

where $T \mid s$ is the triplet set with s as subject in T. Similarly, $(r,o) \in T \mid s$ is the set of all relation-object pair in T. R is the set of all relations and $R\backslash T \mid s$ means all the relations except subject s in T. $o_{\emptyset}$ represents all relations except those in triplet $T \mid s$ will have no corresponding objects.

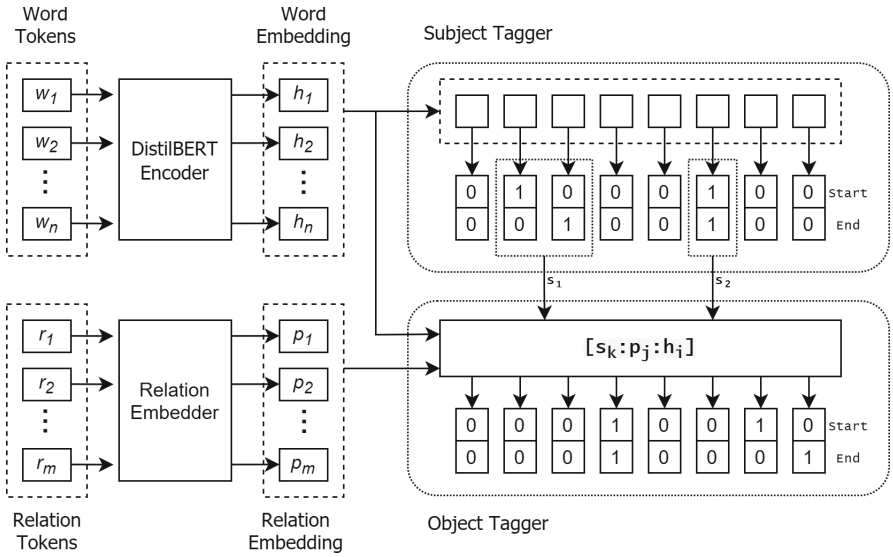

Fig. 2. Architecture of our Relation Extraction model

First, for a given input sentence, a pre-trained DistilBERT encoder is used for extracting tokens for each word and for each predefined relation, an embedding is created as shown in the Eq. 2.

$$\begin{aligned}[h_1, h_2, \cdots h_n] &= E_D([w_1, w_2 \ldots w_n]) \\ [p_1, p_2 \ldots p_m] &= W_r E([r_1, r_2 \ldots r_m]) + b_r\end{aligned} \tag{2}$$

where w_i is word from input sentence and h_i is the output token from DistilBERT encoder E_D Similarly, p_i is the output after relation embedding matrix E embeds predefined relations r_i. W_r and b_r are trainable parameters.

For relation extraction, subject taggers and object taggers are used. The subject tagger defined in Eq. 4 will identify all possible subjects in the word nodes. More specifically, it will tag the head and tail of the subject using sigmoid function, defined in Eq. 3.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

The sigmoid function maps the values between 0 and 1.

$$\begin{aligned}P_i^{ss\_head} &= \sigma\left(W_{s\_head} \text{Tanh}\left(h_i^o\right) + b_{s\_head}\right) \\ P_i^{s_t tail} &= \sigma\left(W_{s\_\text{tail}} \text{Tanh}\left(h_i^o\right) + b_{s\_tail}\right)\end{aligned} \tag{4}$$

where $P_i^{s\_head}$, $P_i^{s\_tail}$ are the probabilities of identifying the ith word as head and tail position of the subject respectively which is calculated by the sigmoid function σ. The values W_{s_head} , W_{s_tail} , $b_{s\_h_{head}}$, b_{s_tail} are trainable weights. h_i^o is the encoded representation of the word from previous stage.

Similarly, the object tagger, defined in Eq. 5 uses encoded word token which is different from token used by subject tagger.

$$\begin{aligned} P_i^{o\_head} &= \sigma\left(W_{o\_head}\bar{h}_{ijk} + b_{o\_head}\right) \\ P_i^{o\_tail} &= \sigma\left(W_{o\_tail}\bar{h}_{ijk} + b_{o\_\text{tail}}\right) \end{aligned} \tag{5}$$

where $P_i^{o\_head}$, $P_i^{o\_tail}$ are the probabilities of identifying the ith word as the head and tail position of the object respectively which is calculated by the sigmoid function σ. The values W_{o_head}, W_{o_tail}, b_{o_head}, b_{o_tail} are trainable weights. The term $\bar{h}_{ijk}$ is encoded word token representation which can be defined as

$$\bar{h}_{ijk} = \text{Tanh}\left(W_h\left[s_k; p_j^0; h_i^0\right] + b_h\right) \tag{6}$$

where s_k is the subject representation of the kth candidate subject, $p^0{}_j$ and $h^o{}_i$ are the encoded representation of the pre-defined relation and word token respectively.

Therefore, in line with Eq. 1, we can define subject tagger and object tagger as Eq. 7 and 8 respectively:

$$P_{\theta_s}(s|x) = \prod_{t\in\{s\_head\,,s\_tail\}}\prod_{i=1}^{N}\left(P_i^t\right)^{\text{I}\{y_i^t=1\}}\left(1-P_i^t\right)^{\text{I}\{y_i^t=0\}} \tag{7}$$

$$P_{\theta_o}(o|x,s,r) = \prod_{t\in\{o\_head\,,o\_tail\}}\prod_{i=1}^{N}\left(P_i^t\right)^{\text{I}\{y_i^t=1\}}\left(1-P_i^t\right)^{\text{I}\{y_i^t=0\}} \tag{8}$$

where θ_s and θ_o are the parameters of the subject tagger and object tagger respectively. $I\{z\} = 1$ if z is true otherwise it is 0 $y_i^{s\_head}$, $y_i^{s\_tail}$ and $y_i^{o\_head}$, $y_i^{o\_tail}$ are binary tags of subject's and object's heads and tails respectively for the ith word in x, For the null object $o_\emptyset$ in Eq. 1, $y_i^{o\varnothing\_head} = y_i^{o\varnothing\_tail} = 0$ for all i.

Taking the logarithm of 1, we get the objective function which is defined in Eq. 9

$$\begin{aligned} L &= \log\prod_{(s,r,o)\in T} p((s,r,o)|x) \\ &= \sum_{s\in T_j}\log p_{\theta_s}(s|x) + \sum_{r\in T_j|s}\log p_{\theta_o}(o|x,s,r) + \sum_{r\in R\setminus T_j|s}\log p_{\theta_o}(o\emptyset|x,s,r) \end{aligned} \tag{9}$$

The log-likelihood function is then maximized by using Stochastic Gradient Descent during training. The learning rate is set as 0.1 for both datasets.

5 Evaluation Metrics

We used precision, recall and F1-scores as evaluation metrics following the baseline approach. A triplet is considered correct only if its predicate and its corresponding subject and object is correct. Additionally, we also used number of trainable parameter in transformer model for comparison as it will help us to identify the efficiency of the model with respect to neural network size as well as gives us an idea of model training time.

6 Implementation Details and Results

The model is implemented with PyTorch library along with CUDA 11. Base DistilBERT model is used from Huggingface [16] with transformer library version 4.12. For both datasets, the models are set to run on maximum of 60 epochs with an early stopping mechanism. The early stopping mechanism will be triggered if there is no improvement in the score for 15 consecutive epochs. Both of the datasets used Stochastic Gradient Boost optimizer with a learning rate of 0.1. The training data is further split into training and validation data. The hyperparameters are determined from this validation data.

We were able to significantly reduce the number of trainable parameters. A comparison of trainable parameters with other transformer-based model is given in Table 3.

Table 3. Comparison of trainable parameters

Model	Parameters
CASREL[9]	107,758,130
RIFRE[10]	117,792,012
Our Model	**67,573,252**

The detailed result of our model from testing of different categories of testing data is tabulated and given in Table 4. It is observable that our model performed fairly good in all triple category scenarios. A slight drop in the score in WebNLG dataset when compared with NYT dataset maybe attributed to the fact that WebNLG main category has most of the data in SEO and EPO form. For the NYT dataset, our model performed the best when there were 4 triples in the sentence and for the WebNLG dataset, the model performed well when there were 3 triples in the sentence. Therefore, from these results, it is evident that our transformer based model is perfectly capable of handling complex scenarios in relational triple extraction.

Table 4. Evaluation results on NYT and WebNLG dataset

	NYT			WebNLG		
	F1-score	Precision	Recall	F1-score	Precision	Recall
Main	89.66	90.22	89.10	88.95	89.86	88.05
Triple-1	87.69	85.46	90.04	85.61	86.26	84.96
Triple-2	90.59	92.60	88.66	88.43	89.22	87.65
Triple-3	91.73	96.14	87.71	91.15	92.35	89.97
Triple-4	93.00	94.11	91.92	88.39	88.27	88.52
Triple-5	88.00	93.66	82.99	90.79	93.52	88.21
Normal	87.46	85.18	89.86	85.66	86.36	84.96
SEO	91.49	94.82	88.39	89.55	90.51	88.61
EPO	91.77	93.65	89.97	90.06	92.77	87.50

7 Conclusion and Future Scope

In this paper, we proposed a light version of transformer-based model for Relation Extraction based on joint entity-relation extraction framework. Our model performed well in all triplet overlapping scenarios such as Entity Pair Overlapping (EPO) and Single Entity Overlapping (SEO) and can extract multiple triplets from same sentence while reducing the number of trainable parameters in the transformer. In the future, we aim to reduce the number of trainable parameters further while improving the performance.

References

1. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long and Short Papers), pp. 4171–4186 (2019)
3. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
4. Zhou, G., Su, J., Zhang, J., Zhang, M.: Exploring various knowledge in relation extraction. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005), pp. 427–434 (2005)
5. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pp. 1003–1011 (2009)
6. Miwa, M., Sasaki, Y.: Modeling joint entity and relation extraction with table representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1858–1869 (2014)
7. Zeng, X., Zeng, D., He, S., Liu, K., Zhao, J.: Extracting relational facts by an end-to-end neural model with copy mechanism. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (vol. 1: Long Papers), pp. 506–514 (2018)
8. Fu, T.-J., Li, P.-H., Ma, W.-Y.: Graphrel: modeling text as relational graphs for joint entity and relation extraction. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1409–1418 (2019)
9. Wei, Z., Su, J., Wang, Y., Tian, Y., Chang, Y.: A novel cascade binary tagging framework for relational triple extraction. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 1476–1488 (2020)
10. Zhao, K., Xu, H., Cheng, Y., Li, X., Gao, K.: Representation iterative fusion based on heterogeneous graph neural network for joint entity and relation extraction. Knowl.-Based Syst. **219**, 106888 (2021)
11. Veena, G., Athulya, S., Shaji, S., Gupta, D.: A graph-based relation extraction method for question answering system. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 944–949. IEEE (2017)
12. Nair, A.M., Bindu, K.R.: Semantic role labelling using transfer learning model. In: Journal of Physics: Conference Series, vol. 1767, p. 012024. IOP Publishing (2021)
13. Gangadharan, V., Gupta, D., Amritha, L., Athira, T.A.: Paraphrase detection using deep neural network based word embedding techniques. In: 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI) (48184), pp. 517–521. IEEE (2020)

14. Riedel, S., Yao, L., McCallum, A.: Modeling relations and their mentions without labeled text. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS (LNAI), vol. 6323, pp. 148–163. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15939-8_10
15. Gardent, C., Shimorina, A., Narayan, S., Perez-Beltrachini, L.: Creating training corpora for NLG micro-planning. In: 55th Annual Meeting of the Association for Computational Linguistics (ACL) (2017)
16. Wolf, T., et al.: Huggingface's transformers: state-of-the-art natural language processing. arXiv preprint arXiv:1910.03771 (2019)