NLP: Relation Extraction

Nazgul Mamasheva

1 Introduction

Relation extraction is a natural language processing (NLP) task that involves identifying and classifying relationships between entities mentioned in a text. The goal is to extract structured information about how different entities are connected or related to each other within a given context.

In the given text "The capital of Italy is Rome.", the goal of relation extraction would be to identify and classify the relationship "Capital" between the entities "Italy" and "Rome". The output could be a structured representation like (Italy, Capital, Rome).

2 Implementation

Our approach for Relation extraction is consists of two-phase pipelines: Phase 1 - Entity recognition and Phase 2 - Relation classification.

2.1 Phase 1 - Entity Recognition (ER)

Entity Recognition, also known as Named Entity Recognition (NER) is a natural language processing (NLP) task that involves identifying and classifying entities (such as names of people, organizations, locations, and other specific types of information) in unstructured text.

In the initial phase of our pipeline, we focus on Entity Recognition task where our objective is to identify entities without necessarily classifying them.

For each sentence we predict entities by trained model. From predicted entities we form all possible combinations of two entities, where the first entity represents the subject and the second one the object. Thus, we build the input data for next phase by forming relation tuple candidates.

2.1.1 Data processing

From dataset we can see that the entities may consist of one or several subsequent words, therefore we need to store whole span of entities words. In the given dataset the entities are indicated as subject or object along with their start and end positions. So, we retrieve and store start and end indices of entities.

For our entity recognition task we don't need predefined labels vocabulary, since we don't classify entities. So, the entities are not assigned specific labels such as *PER*, *LOC*, *ORG*, etc. Instead, all entities are uniformly labeled as *ENT* (short for Entity) and converted into BIO format. The BIO format distinguishes the beginning (*B-ENT*) and inside (*I-ENT*) parts of entities, along with the outside (*O*) label for non-entity words. This way of labeling allows us to recognize entities that consist of more than one word.

As illustrated in Figure 1, we set the label for each token in the text and we set *B-ENT* or *I-ENT* for the first token of each entity word, for all other tokens that are not entities we set label *O*. The text is processed, and each word or token is labeled with its corresponding labels. The output of this process is a sequence of labeled tokens.

2.1.2 Model

As a model for our first phase task we chose BertForTokenClassification model. BertForTokenClassification is a pre-trained transformer-based model, part of the BERT (Bidirectional Encoder Representations from Transformers) family, specifically designed for token-level classification tasks. Token classification involves assigning a label to each token in a sequence. This model is fine-tuned for tasks like Named Entity Recognition (NER) or any other sequence labeling task where each token needs to be classified into predefined categories. BertForTokenClassification has a token-level classification head on top of the BERT base, allowing it to handle tasks that require recognizing and classifying entities or labels for each token in a sequence. It is commonly used for tasks where the boundaries of entities or labels are not known beforehand, and

the model needs to learn to predict them. Thus, BertForTokenClassification is best suited for our Entity recognition task.

The model dbmdz/bert-large-cased-finetuned-conll03-english is a fine-tuned version of the BERT model trained specifically for the task of named entity recognition (NER) on the CoNLL-2003 dataset for English.

2.2 Phase 2 - Relation Classification (RC)

Relation classification is a natural language processing (NLP) task that involves determining the semantic relationship between entities mentioned in a text. In this task, the goal is to classify the type of relationship that exists between two entities. This helps in understanding the connections and associations between different entities in a given context.

After identifying entities and preparing relation tuple candidates, the next step is to classify relationships between pairs of entities in the text. To distinguish which entity is subject and which one is an object in a sentence, we add special tags <SUBJ> and </SUBJ> for a subject and <OBJ> and </OBJ> for an object. In this way we create new sentences with appended special tags and prepare input data for relation classification model.

If in the first phase we obtained and prepared input data from a given data, in the second phase we modify sentences by adding tags. These tags help our model to recognise and differentiate the subject and object entities.

2.2.1 Data processing

In the second phase we need predefined labels vocabulary. So, we retrieve list of all possible relation labels from a given relation2id file and we pre-define label2id and id2label mapping dictionaries and pass them to our models. The label2id is a mapping from label names to numerical values, and id2label is the reverse mapping from numerical values to label names. These mappings are essential for converting between the textual representation of labels and the numerical format that the model understands during training and inference.

The relations are the relationships between entities in the sentences. We processing relations by iterating over each sentence and its corresponding relations. As illustrated in Figure 2, for each relation in the sentence we locate the subject and object entities within the tokens and surrounds them with special tags ("<SUBJ>" and "</SUBJ>" for subjects, "<OBJ>" and "</OBJ>" for objects).

As a final data processed we use a tokenizer to tokenize the modified sentences and pads them to ensure uniform length within the batch.

2.2.2 Model

BertForSequenceClassification is a pre-trained BERT model fine-tuned specifically for sequence classification tasks. It is designed for tasks where the goal is to classify an entire sequence or sentence into one or more predefined categories or labels.

In our relation classification, we perform classification on a sequence of words in a sentence, so BertForSequenceClassification is well suited for our task.

3 Trainer

AutoTokenizer is preferable for its model-agnostic nature, enabling easy switching between different pre-trained models. So, we used AutoTokenizer for both our pipeline's models.

As shown on Table 1, we chose the batch size as 8, with training number as 5, these values are optimal for our models training. As an optimizer we used Hugging Face's AdamW. AdamW is an optimizer specifically designed for training deep learning models. It's an extension of the popular Adam optimizer and includes a weight decay fix. The "W" in AdamW stands for "Weight Decay." This optimizer is commonly used with transformer-based models like BERT to improve training stability and convergence.

4 Results

We can't check the results of two phases separately, since we'll not be given test dataset for entity recognition task during testing and also we'll not be given a separate train dataset for the relation classification task, instead we'll create input data to second phase from first phase's result. So, we can only obtain the combined final result of two tasks. So, as a final result we get F1 score as is a 0.5212, precision as a 0.3982 and recall is 0.7542.

5 Conclusion

Our approach demonstrates the effectiveness of leveraging pre-trained transformer models for relation extraction tasks, emphasizing the importance of entity recognition and relation classification in understanding textual relationships.

Hyperparameter	Value
Batch size	8
Training number	5
Optimizer	AdamW

Table 1: Hyperparameters and their values.

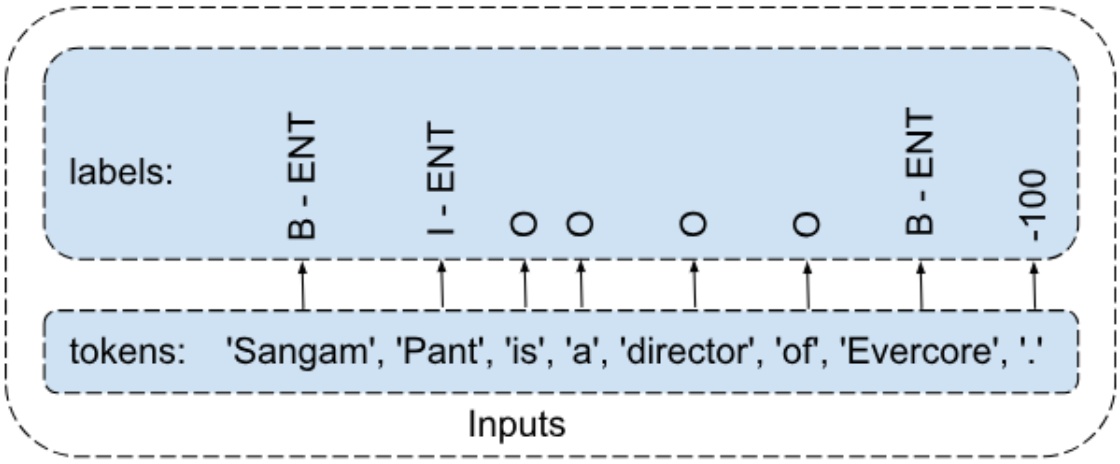

Figure 1: Input scheme of Entity Recognition.

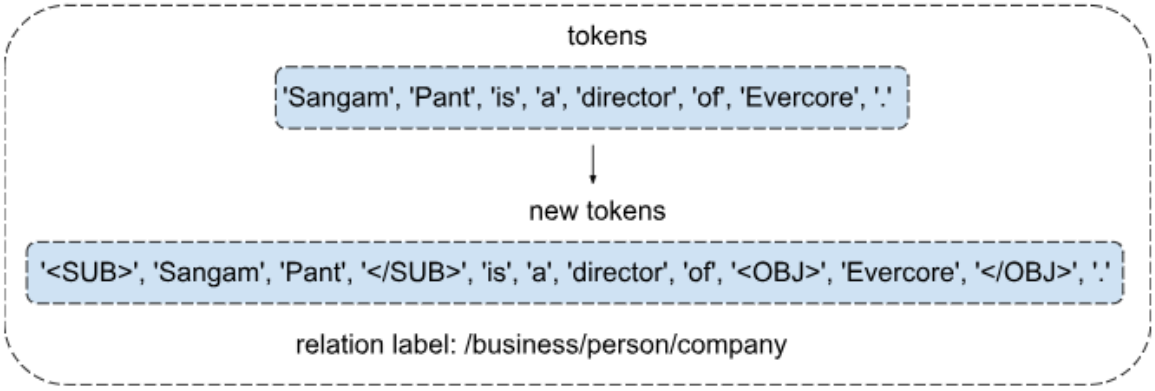

Figure 2: Input scheme of Relation Classification.