An Automated Named Entity Recognition System from Text

Abstract

By

Anshu Kumar

Enrollment no: 12017002002022

Under the Supervision of

PROF. ANUPAM MONDAL

Department of Computer Science & Engineering

Institute of Engineering & Management

West Bengal, India

September, 2020

Abstract

Named entity recognition (NER) — sometimes referred to as entity chunking, extraction, or identification — is the task of identifying and categorizing key information (entities) in text. An entity can be any word or series of words that consistently refers to the same thing. Every detected entity is classified into a predetermined category. For example, an NER machine learning (ML) model might detect the word “super.AI” in a text and classify it as a “Company”.

NER is a form of natural language processing (NLP), a subfield of artificial intelligence. NLP is concerned with computers processing and analyzing natural language, i.e., any language that has developed naturally, rather than artificially, such as with computer coding languages.

At the heart of any NER model is a two step process:

1. Detect a named entity
2. Categorize the entity

Beneath this lie a couple of things.

Step one involves detecting a word or string of words that form an entity. Each word represents a token: “The Great Lakes” is a string of three tokens that represents one entity. Inside-outside-beginning tagging is a common way of indicating where entities begin and end. We’ll explore this further in a future blog post. In order to learn what is and is not a relevant entity and how to categorize them, a model requires training data. The more relevant that training data is to the task, the more accurate the model will be at completing said task. Train your model on Victorian gothic literature, and it will probably struggle to navigate Twitter.

Once you have defined your entities and your categories, you can use these to label data and create a training dataset (our named entity recognition data program can do this for you automatically). You then use this training dataset to train an algorithm to label your text predicatively.

NER is suited to any situation in which a high-level overview of a large quantity

of text is helpful. With NER, you can, at a glance, understand the subject or theme of a body of text and quickly group texts based on their relevancy or similarity.

Some notable NER use cases include:
- **Human resources:** Speed up the hiring process by summarizing applicants' CVs; improve internal workflows by categorizing employee complaints and questions.

- **Customer support:** Improve response times by categorizing user requests, complaints and questions and filtering by priority keywords. Search and recommendation engines improve the speed and relevance of search results and recommendations by summarizing descriptive text, reviews, and discussions.

- **Content classification:** Surface content more easily and gain insights into trends by identifying the subjects and themes of blog posts and news articles.

- **Health care:** Improve patient care standards and reduce workloads by extracting essential information from lab reports.

- **Academia:** Enable students and researchers to find relevant material faster by summarizing papers and archive material and highlighting key terms, topics, and themes.The EU's digital platform for cultural heritage, Europeana, is using NER to make historical newspapers searchable.

If you think that your business or project could benefit from NER, it's pretty easy to start out. There are a number of excellent open-source libraries that can get you going, including NLTK, SpaCy, and Stanford NER. Each has its own pros and cons, which we'll be exploring in more detail soon.

But before you begin using one of these libraries to build a model, you will need to produce a relevant labeled dataset to train the model on. That's where super.AI is there to help. Using our named entity recognition data program, you provide us your raw text and desired entities and categories. We'll label the text you send and return a high quality training dataset that you can take to train and tailor your NER model.

References:

[1] A. Berger, S. Della Pietra, and V. Della Pietra. 1996. A maximum entropy approach to natural language processing. Computational Linguistics, 22(1):39–71.

[2] Daniel M. Bikel, Richard L. Schwartz, and Ralph M. Weischedel. 1999. An algorithm that learns what's in a name. Machine Learning, 34(1-3):211–231.

[3] A. Borthwick. 1999. A Maximum Entropy Approach to Named Entity Recognition. Ph.D. thesis, New York University.

[4] E. Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. Computational Linguistics, 21(4):543–565.

[5] J. N. Darroch and D. Ratcliff. 1972. Generalized iterative scaling for log-linear models. The Annals of Mathematical Statistics, 43(5):1470–1480.

SIGNATURE OF STUDENTS:

1. Anshu Kumar

SIGNATURE OF MENTOR(S):

Anupam Mondal

DATE: 03/09/2020