Name:Kaushik Kotian Roll No.:14 Div:D20B

AIML LAB

EXPERIMENT 5

Aim : Natural language Entity Extraction from medical reports.

Theory :

What is Natural Language Processing (NLP)?

Natural Language Processing (NLP) is a branch of artificial intelligence that focuses on the interaction between computers and human (natural) languages. The goal of NLP is to enable computers to understand, interpret, and generate human language in a way that is both meaningful and useful. NLP encompasses various tasks such as text analysis, speech recognition, language translation, sentiment analysis, and entity recognition.

2. Different Techniques Used in NLP

2.1 Rule-Based Techniques

Rule-based NLP systems rely on manually created linguistic rules. These rules are defined by domain experts to handle specific language processing tasks.

Examples:

- Regular Expressions for text pattern matching.
- Tokenization rules for breaking down sentences into words.
- Named Entity Recognition (NER) using predefined dictionaries or pattern matching.

Advantages:

- Simple and effective for well-defined tasks.
- Offers high precision for specific use cases with known patterns.

Challenges:

- Limited scalability and generalization.
- Manually intensive and difficult to maintain for large datasets.

2.2 Statistical Techniques using Machine Learning Models

Statistical NLP models are trained on large datasets to learn patterns and make predictions based on the statistical properties of the text.

Examples:

- Naive Bayes for text classification.
- Support Vector Machines (SVM) for sentiment analysis.
- Hidden Markov Models (HMM) for part-of-speech tagging.
- Conditional Random Fields (CRF) for Named Entity Recognition (NER).

Advantages:

- Can handle a wide variety of tasks by learning from data.
- More adaptable and scalable than rule-based systems.

Challenges:

- Requires large, labeled datasets for training.
- May suffer from poor interpretability (black-box models).

2.3 Transfer Learning

Transfer learning involves leveraging pre-trained models on one task and fine-tuning them on a specific NLP task. This technique reduces the need for large labeled datasets by reusing knowledge from related tasks.

Examples:

- BERT (Bidirectional Encoder Representations from Transformers): Pre-trained on vast amounts of text data and can be fine-tuned for tasks like text classification and question answering.
- GPT (Generative Pretrained Transformer): Used for text generation and dialogue systems.

Advantages:

- Reduces training time and data requirements.
- Achieves state-of-the-art performance on many NLP tasks.

Challenges:

- Requires computational resources for fine-tuning.
- May exhibit bias if the pre-trained models have been trained on biased datasets.

3. Various NLP Libraries and Their Frameworks

NLTK (Natural Language Toolkit): A popular library in Python used for text processing, tokenization, stemming, lemmatization, and basic NLP tasks.

SpaCy: An industrial-strength NLP library known for its fast processing and support for deep learning. Used for named entity recognition, part-of-speech tagging, and dependency parsing.

Hugging Face Transformers: A library providing state-of-the-art pre-trained models like BERT, GPT, and RoBERTa for NLP tasks such as text classification, translation, and question answering.

Stanford NLP: A library for complex NLP tasks such as parsing, named entity recognition, and coreference resolution.

Gensim: A Python library focused on unsupervised topic modeling and document similarity using techniques like Word2Vec and TF-IDF.

4. What are Large Language Models (LLM)?

Large Language Models (LLMs) are advanced NLP models that are trained on vast amounts of text data to understand and generate human-like language. They typically use deep learning architectures like Transformers to capture context, semantics, and syntax. LLMs are often pre-trained on general language tasks and then fine-tuned for specific applications.

Examples:

- GPT (Generative Pretrained Transformer): Known for generating coherent and contextually accurate text.
- BERT (Bidirectional Encoder Representations from Transformers): Used for understanding the context of words in a sentence and excelling in tasks like question answering and text classification.
- T5 (Text-to-Text Transfer Transformer): Converts every NLP task into a text-to-text framework, which is powerful for tasks like summarization, translation, and text generation.

Advantages:

- High accuracy and versatility across many NLP tasks.
- Capable of zero-shot and few-shot learning, where they perform well even with limited task-specific training data.

5. NLP Libraries Used in Healthcare Industry

The healthcare industry uses NLP for various tasks such as extracting information from clinical notes, medical reports, and prescriptions. The following libraries are commonly used:

MedSpaCy: A healthcare extension of the SpaCy library, customized for processing medical text, performing entity recognition, and negation detection in clinical narratives.

Hugging Face's Transformers: Used in healthcare to leverage pre-trained models for clinical document processing, especially BERT-based models like BioBERT and ClinicalBERT, which are trained on biomedical literature.

PyText: An NLP framework by Facebook used for text classification and entity extraction in healthcare-specific tasks.

These NLP tools are applied to extract entities like patient demographics, medical conditions, treatments, and outcomes from medical reports and clinical notes, aiding in tasks like clinical decision support and patient care analysis.

Stanza is an open-source Natural Language Processing (NLP) library developed by the Stanford NLP Group. It provides a set of easy-to-use tools for various NLP tasks, with a focus on high-quality performance across many languages. Stanza is designed to handle a wide range of NLP tasks including:

- Tokenization (splitting text into words or tokens)
- Part-of-speech (POS) tagging (identifying the grammatical category of each token)
- Lemmatization (converting words to their base form)
- Named Entity Recognition (NER)

OUTPUT:

```
import stanza
# download and initialize the CRAFT pipeline
stanza.download('en', package='craft')
nlp = stanza.Pipeline('en', package='craft')
# annotate example text
doc = nlp('He had a sore throat.')
# print out dependency tree
doc.sentences[0].print_dependencies()
```

```
  checkpoint = torch.load(filename, lam
INFO:stanza:Done loading processors!
('He', 2, 'nsubj')
('had', 0, 'root')
('a', 5, 'nummod')
('sore', 5, 'amod')
('throat', 2, 'obj')
('.', 2, 'punct')
```

('He', 2, 'nsubj')

- **"He"** is the **subject** of the verb "had."
- **2** means "He" depends on the 2nd word, which is "had."
- **nsubj** stands for **nominal subject**: "He" is the subject of the verb "had."

('had', 0, 'root')

- **"had"** is the main verb of the sentence and the **root** of the dependency tree.
- **0** means that "had" has no head because it's the root of the sentence.

('a', 5, 'nummod')

- **"a"** modifies the noun **"throat."**
- **5** means "a" is dependent on the 5th word, which is "throat."
- **nummod** stands for **numeral modifier**, typically used for quantifiers like "a", "two", etc.

('sore', 5, 'amod')

- **"sore"** is an **adjectival modifier** describing the noun "throat."
- **5** means "sore" depends on the 5th word, which is "throat."
- **amod** stands for **adjective modifier**, meaning "sore" is modifying "throat."

('throat', 2, 'obj')

- **"throat"** is the **object** of the verb "had."
- **2** means "throat" depends on the 2nd word, which is "had."
- **obj** stands for **object**: "throat" is the object that "He had."

('.', 2, 'punct')

- **"."** is the punctuation mark (period) at the end of the sentence.

- **2** means the period is dependent on the 2nd word, which is "had."
- **punct** stands for **punctuation**.

```
nlp = stanza.Pipeline('en', package=None, processors={'tokenize':
'spacy', 'ner': 'i2b2'}, tokenize_pretokenized=True)
#The code is missing a tokenizer. Add the spacy tokenizer to the
processors dictionary.
# annotate pre-tokenized sentences
doc = nlp([['John', 'was', 'prescribed', 'two', 'tablets', 'of',
'Ibuprofen','.']])
```

	id	text	misc	start_char	end_char	ner	multi_ner
0	1	John		0	4	O	(O,)
1	2	was		5	8	O	(O,)
2	3	prescribed		9	19	O	(O,)
3	4	two		20	23	O	(O,)
4	5	tablets		24	31	O	(O,)
5	6	of		32	34	O	(O,)
6	7	Ibuprofen		35	44	S-TREATMENT	(S-TREATMENT,)
7	8	.	SpaceAfter=No	45	46	O	(O,)

Entities:

- Each word in the sentence is annotated with a ner tag and other metadata like id, text, start_char, end_char, etc.
- ner: This represents the Named Entity tag, which indicates what type of entity (if any) a word represents.

NER Tags:

- O: This indicates that the word is not part of any named entity (i.e., it's "Outside" of a relevant entity).
- S-TREATMENT: The word "Ibuprofen" has this tag, meaning it is identified as a "Treatment", a specific type of medical entity.