

Extracción de entidades con Rasa

Miguel Edo Goterris



TramTime

- Quiero ir desde la estación de Benimaclet a la de Universidad Politécnica.
- De La carrasca a Tossal del rei

Benimaclet -> Universitat Politècnica:

Linea: 6	Hora: 05:55:00	Destino: Marítim
Linea: 4	Hora: 06:05:00	Destino: Dr. Lluch
Linea: 4	Hora: 06:25:00	Destino: Dr. Lluch
Linea: 6	Hora: 06:28:00	Destino: Marítim

⚠ Problema: extracción de entidades

- Dado un mensaje, extraer la **información relevante**.
- Fundamental para crear chatbots que usen **lenguaje natural**.
- Dependiendo de como se elaboren los mensajes puede ser **complicado**.

Entidades y roles

Dos tipos de datos distintos.

```
entities:  
  - departure_station  
  - destination_station
```

Un tipo de datos, dos funciones.

```
entities:  
  - station:  
    roles:  
      - departure  
      - destination
```

Expresiones regulares y look up tables

- regex: account_number
examples: |
 - \d{10,12}

Formatos fijos:

- DNI
- Matrículas
- Códigos postales


- lookup: station
examples: |
 - À Punt
 - Aeroport
 - ...
 - Vicente Zaragozá
 - Xàtiva

Expresiones regulares y look up tables

Rather than directly returning matches, these lookup tables work by marking tokens in the training data to indicate whether they've been matched. This provides an extra set of features to the conditional random field entity extractor (`ner_crf`) This lets you identify entities that haven't been seen in the training data and also eliminates the need for any post-processing of the results.

Sinónimos

- `synonym: Universitat Politècnica`
`examples: |`
 - El poli
 - La UPV
 - La politècnica
 - Politècnica

- Distintas formas de referirse a una misma entidad.
- Rasa asignará `Universitat Politècnica` si extrae cualquiera de las entidades de ejemplo.
-  TramTime no usa.

Intents

Quiero ir desde Benimaclet a la Universitat Politècnica.

```
Quiero ir desde [Benimaclet]
{"entity": "o_station", "value": "benimaclet"}
hasta [Universitat politècnica]
{"entity": "d_station", "value": "Universitat politècnica"}
```

Quiero ir de beni al poli.

```
Quiero ir de [beni]
{"entity": "o_station", "value": "benimaclet"}
al [poli]
{"entity": "d_station", "value": "Universitat politècnica"}
```


Intents

Quiero ir de **beni** al **poli**.

Quiero ir desde

```
[Benimaclet>{"entity": "station", "role": "departure"}
```

hasta

```
[La politecnica>{"entity": "station", "role": "destination"}
```

- Origen: **beni**
- Destino: **poli**

Más intents

Generar intents automáticamente no es una mala idea.

```
import random
frase = "- Voy de {origen} a {destino}"
estaciones = [...]
origen, destino = random.sample(estaciones, 2)
origen = f'[{origen}][{"entity": "station", "role": "departure"}]'
destino = f'[{destino}][{"entity": "station", "role": "destination"}]'
frase.format(origen=origen, destino=destino)
```

Pipeline

```
pipeline:
- name: SpacyNLP
  model: es_core_news_sm
- name: SpacyTokenizer
- name: SpacyFeaturizer
  pooling: mean
- name: LexicalSyntacticFeaturizer
- name: CountVectorsFeaturizer
  analyzer: char_wb
  min_ngram: 1
  max_ngram: 4
- name: DIETClassifier
  epochs: 100
- name: SpacyEntityExtractor
```

1. Tokenizado
2. Embeddings preentrenados.
3. Extracción de características léxicas y sintácticas.
4. Dual Intent and Entity Transformer. Usa embeddings.
5. BILOU con modelos preentrenados.

Postprocesado

Las entidades extraídas se deben convertir en IDs para acceder a la API.

- Sinonimos.
- Distancia de Damerau-Levenshtein.

Ejemplos

- me gustaria ir de trinitat hasta la estación de valencia sud ✓
- de trinitat a pont de fusta ✓
- anire de beniferro a beninanet ✓
- ire de beni a la politecnica ?
- quiero llegar a betera saliendo de betero ?

La calidad del bot depende de nuestro conocimiento. Almacenando las consultas y mejorando el bot en base a estas se puede conseguir una muy buena extracción de entidades.