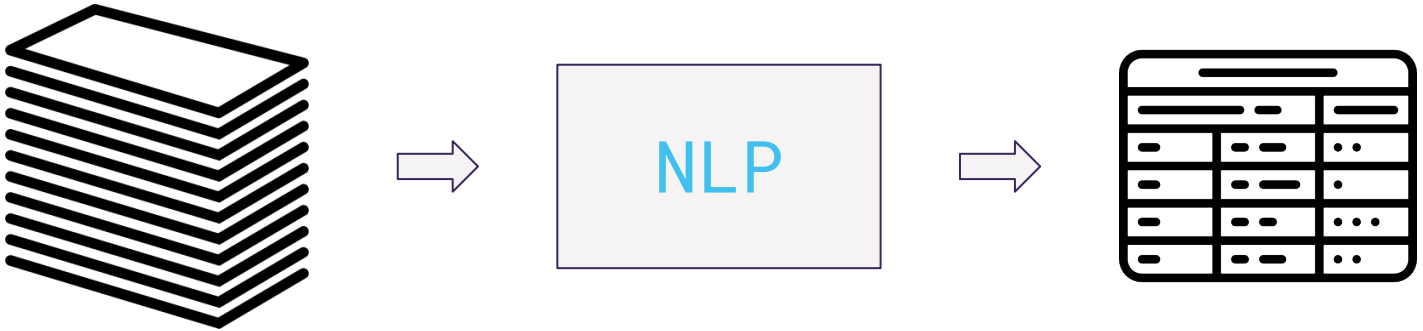
sages

Ekstrakcja informacji

(ang. information extraction)

Czym jest ekstrakcja informacji?

Zadaniem **ekstrakcji informacji** jest automatyczne **wydobycie**, **strukturalizacja i interpretacja** danych z tekstu przy użyciu metod i narzędzi do przetwarzania języka naturalnego.



Ekstrakcja, a wyszukiwanie informacji

Podstawową różnicą między ekstrakcją, a wyszukiwaniem informacji jest **postać oczekiwanego wyniku**.

Ile wynosiły dywidendy za akcje spółek notowanych na GPW?

Wyszukiwanie

Ekstrakcja

Spółka	Dywidenda za akcję	Rok
KERNEL	1,59 zł	2021
TIM	1,20 zł	2020
ARCHICOM	2,53 zł	2020

Zadania ekstrakcji informacji

- Rozpoznawanie jednostek:
  - jednostki nazewniczych (NER),
  - wyrażenia temporalne, liczbowe,
- Relacje między elementami (aspekt statyczny):
  - położenia — X ma-siedzibę-w Y,
  - przynależności — X jest-pracownikiem Y, X jest-prezesem Y,
- Wydarzenia — kto, co, kiedy, kogo, ...? (aspekt dynamiczny):
  - zmiana położenia — X przeniósł-siedzibę-do Y,
  - zmiana stanowiska — X awansował-na-kierownika-w Y,
- Wypełnianie szablonów:
  - łączy w sobie rozpoznawanie elementów, relacji oraz wydarzeń,
  - każdy rekord zawiera określona liczbę pól do wypełnienia,
  - np. umowa kupna-sprzedaży nieruchomości: dane osobowe kupującego i sprzedającego, adres nieruchomości, kwota sprzedaży, itd.

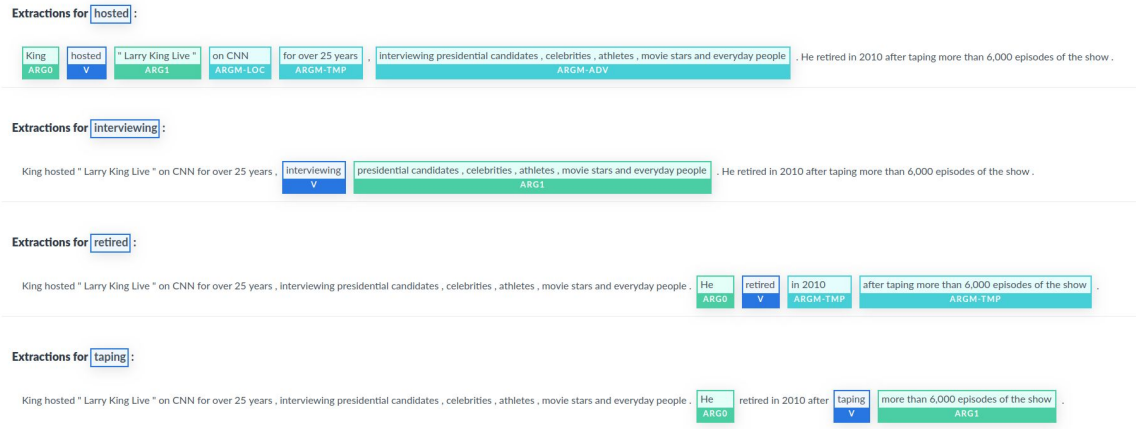
Przykładowe projekty

AllenNLP Demo

Rozpoznawanie zdarzeń

https://demo.allennlp.org/open-information-extraction/open-information-extraction

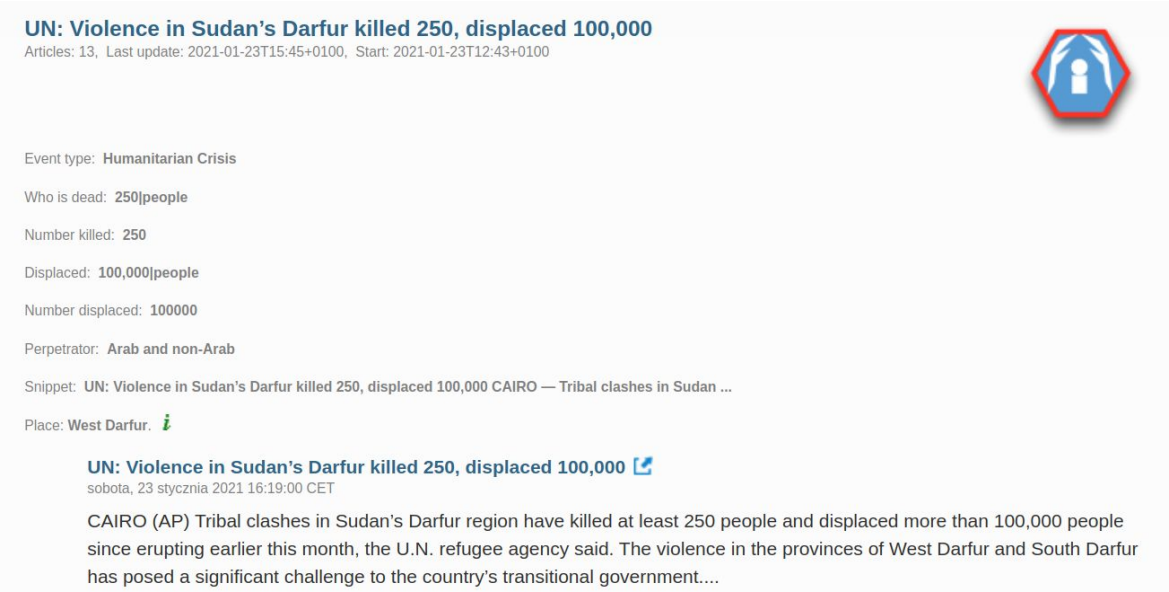
King hosted "Larry King Live" on CNN for over 25 years, interviewing presidential candidates, celebrities, athletes, movie stars and everyday people. He retired in 2010 after taping more than 6,000 episodes of the show.



News Brief

Wypełnianie szablonów

https://emm.newsbrief.eu/NewsBrief/eventedition/pl/latest_en.html

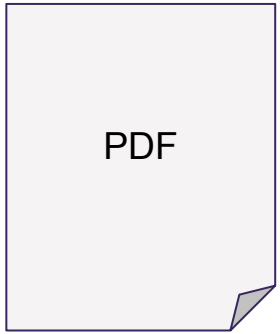


PolEval 2020 Task 4

Wypełnianie szablonów

http://poleval.pl/tasks/task4

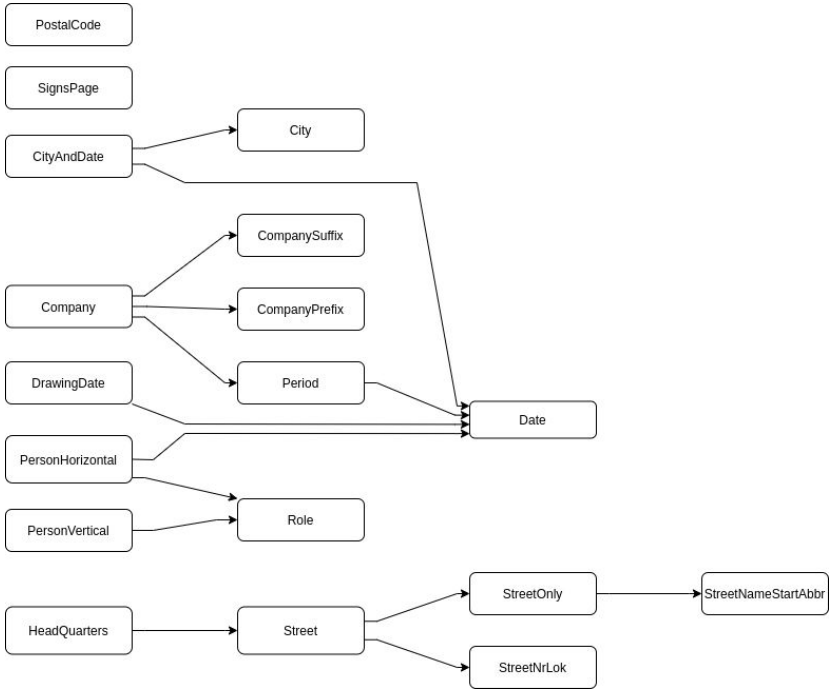
Ekstrakcja informacji z raportów finansowych spółek giełdowych



- nazwa spółki,
- data sporządzenia raportu,
- okres, którego dotyczy sprawozdanie,
- adres siedziby spółki,
- dane osób podpisanych pod raportem
  - imię i nazwisko,
  - stanowisko w spółce,
  - data złożenia podpisu.

PolEval 2020 Task 4 — CLEX

- https://github.com/CLARIN-PL/CLEX
- implementacja w Javie,
- system regułowy,
- wykorzystuje kaskadę reguł do rozpoznawania coraz bardziej złożonych elementów,
- dwa typy reguł:
  - wykrywanie informacji cząstkowych,
  - agregacja informacji cząstkowych.



Ekstrakcja informacji w praktyce

Ekstrakcja informacji w praktyce

- Ze względu na dużą różnorodność zadań ekstrakcji informacji każde zadanie **wymaga indywidualnego podejścia**.
- Informacje wydobywane są na podstawie **przesłanek** i **sygnałów** występujących w treści dokumentu — **słowa kluczowe**, **konstrukcje językowe**, **szablony**.
- W zależności od zadania należy określić, które istotne elementy **są już rozpoznawane i obsługiwane** przez obecne moduły.
- Dla elementów, które nie są rozpoznawane należy podjąć decyzję, w jaki sposób można je rozpoznać — **słownikowo**, **regułowo** czy też z użyciem **metod maszynowego uczenia**.

Przykładowe zadanie

- definicja zadania
  - *Ile wynoszą dywidendy za akcje spółek notowanych na GPW?*
  - pola:
    - nazwa spółki,
    - kwota dywidendy za akcję,
    - rok,
- określenie źródła danych
  - tytuły wiadomości z portalu informacyjnego dla inwestorów,
  - Przykłady:
    - Apator wypłaci 1,20 zł dywidendy i skupi akcje własne za 20 mln zł
    - Aplisens wypłaci 0,26 zł dywidendy na akcję
    - APS Energia chce wypłacić 4 gr dywidendy na akcję
- analiza danych
  - nazwa spółki — rozpoznana przez moduł NER,
  - kwota — nie jest rozpoznawana przez moduł NER, wymaga dodania dedykowanego modułu,
  - rok — nie występuje w nagłówkach, ale można pobrać z metadanych wiadomości.
  - wzorce
    - **[orgName]** wypłaci **[money]** dywidendy,
    - **[orgName]** chce wypłacić **[money]** dywidendy.

Ekstrakcja informacji w spaCy

Matcher — wzorce sekwencji tokenów

https://spacy.io/usage/rule-based-matching#matcher

```
import spacy
from spacy.matcher import Matcher

nlp = spacy.load("en_core_web_sm")
matcher = Matcher(nlp.vocab)
# Add match ID "HelloWorld" with no callback and one pattern
pattern = [{"LOWER": "hello"}, {"IS_PUNCT": True}, {"LOWER": "world"}]
matcher.add("HelloWorld", None, pattern)

doc = nlp("Hello, world! Hello world!")
matches = matcher(doc)
for match_id, start, end in matches:
    string_id = nlp.vocab.strings[match_id]  # Get string representation
    span = doc[start:end]  # The matched span
    print(match_id, string_id, start, end, span.text)
```

Matcher — wzorce sekwencji tokenów

Atrybuty

ATTRIBUTE	VALUE TYPE	DESCRIPTION
`ORTH`	unicode	The exact verbatim text of a token.
`TEXT` V2.1	unicode	The exact verbatim text of a token.
`LOWER`	unicode	The lowercase form of the token text.
`LENGTH`	int	The length of the token text.
`IS_ALPHA`, `IS_ASCII`, `IS_DIGIT`	bool	Token text consists of alphabetic characters, ASCII characters, digits.
`IS_LOWER`, `IS_UPPER`, `IS_TITLE`	bool	Token text is in lowercase, uppercase, titlecase.
`IS_PUNCT`, `IS_SPACE`, `IS_STOP`	bool	Token is punctuation, whitespace, stop word.
`IS_SENT_START`	bool	Token is start of sentence.
`SPACY`	bool	Token has a trailing space.
`LIKE_NUM`, `LIKE_URL`, `LIKE_EMAIL`	bool	Token text resembles a number, URL, email.
`POS`, `TAG`, `DEP`, `LEMMA`, `SHAPE`	unicode	The token's simple and extended part-of-speech tag, dependency label, lemma, shape. Note that the values of these attributes are case-sensitive. For a list of available part-of-speech tags and dependency labels, see the Annotation Specifications .
`ENT_TYPE`	unicode	The token's entity label.

Wyrażenia regularne

```
# Match different spellings of token texts
pattern = [{"TEXT": {"REGEX": "deff?in[ia]tely"}}]

# Match tokens with fine-grained POS tags starting with 'V'
pattern = [{"TAG": {"REGEX": "^V"}}]
```

Opcjonalność, powtórzenia

```
pattern = [{"LOWER": "hello"},
           {"IS_PUNCT": True, "OP": "?"}]
```

OP	DESCRIPTION
`!`	Negate the pattern, by requiring it to match exactly 0 times.
`?`	Make the pattern optional, by allowing it to match 0 or 1 times.
`+`	Require the pattern to match 1 or more times.
`*`	Allow the pattern to match zero or more times.

EntityRuler — regułowa anotacja

https://spacy.io/usage/rule-based-matching#entityruler

```
from spacy.lang.en import English
from spacy.pipeline import EntityRuler

nlp = English()
ruler = EntityRuler(nlp)
patterns = [{"label": "ORG", "pattern": "Apple"},
            {"label": "GPE", "pattern": [{"LOWER": "san"}, {"LOWER": "francisco"}]}]
ruler.add_patterns(patterns)
nlp.add_pipe(ruler)

doc = nlp("Apple is opening its first big office in San Francisco.")
print([(ent.text, ent.label_) for ent in doc.ents])
```

RUN

```
[('Apple', 'ORG'), ('San Francisco', 'GPE')]
```

Doc.retokenizer

- służy do łączenia i dzielenia tokenów,
- zmiany w tokenizacji zostają wprowadzone po opuszczeniu kontekstu.

```
doc = nlp("I like David Bowie")
with doc.retokenize() as retokenizer:
    attrs = {"LEMMA": "David Bowie"}
    retokenizer.merge(doc[2:4], attrs=attrs)
```

Przydatne moduły

- **tagger**
  - dopasowanie sekwencji tokenów,
  - dopasowanie po części mowy (np. czasowniki) i lematach,
- **ner** — rozpoznanie i dopasowanie jednostek nazewniczych po kategoriach,
- **parser** — wykorzystanie relacji zależnościowych do definiowania generycznych reguł, które uwzględniają dodatkowe elementy w wypowiedzi.