Evaluate Presidio Analyzer using the Presidio Evaluator framework

This notebook demonstrates how to evaluate a Presidio instance using the presidio-evaluator framework Steps:

1. Load dataset from file
2. Simple dataset statistics
3. Define the AnalyzerEngine object (and its parameters)
4. Align the dataset's entities to Presidio's entities
5. Set up the Evaluator object
6. Run experiment
7. Evaluate results
8. Error analysis

For an example with a custom Presidio instance, see notebook 5.

In [1]:
```
# install presidio evaluator via pip if not yet installed

#!pip install presidio-evaluator
```

In [2]:
```
from pathlib import Path
from pprint import pprint
from collections import Counter
from typing import Dict, List
import json

from presidio_evaluator import InputSample
from presidio_evaluator.evaluation import SpanEvaluator, ModelError, Plotter
from presidio_evaluator.experiment_tracking import get_experiment_tracker

import pandas as pd

pd.set_option("display.max_columns", None)
pd.set_option("display.max_rows", None)
pd.set_option("display.max_colwidth", None)
```

```
%reload_ext autoreload
%autoreload 2
```

```
stanza and spacy_stanza are not installed
Flair is not installed by default
```

1. Load dataset from file

In [3]:
```
dataset_name = "synth_dataset_v2.json"
dataset = InputSample.read_dataset_json(Path(Path.cwd().parent, "data", dataset_name))
print(len(dataset))
```

```
tokenizing input:   0%|          | 0/1500 [00:00<?, ?it/s]
loading model en_core_web_sm
tokenizing input: 100%|██████████| 1500/1500 [00:06<00:00, 229.51it/s]
1500
```

This dataset was auto generated. See more info here Synthetic data generation.

In [4]:
```
def get_entity_counts(dataset: List[InputSample]) -> Dict:
    """Return a dictionary with counter per entity type."""
    entity_counter = Counter()
    for sample in dataset:
        for tag in sample.tags:
            entity_counter[tag] += 1
    return entity_counter
```

2. Simple dataset statistics

In [5]:
```
entity_counts = get_entity_counts(dataset)
print("Count per entity:")
pprint(entity_counts.most_common(), compact=True)

print("\nMin and max number of tokens in dataset: "\
f"Min: {min([len(sample.tokens) for sample in dataset])}, "\
f"Max: {max([len(sample.tokens) for sample in dataset])}")

print(f"Min and max sentence length in dataset: " \
```

```
f"Min: {min([len(sample.full_text) for sample in dataset])}, "\
f"Max: {max([len(sample.full_text) for sample in dataset])}")

print("\nExample InputSample:")
print(dataset[0])
```

```
Count per entity:
[('O', 19626), ('STREET_ADDRESS', 3071), ('PERSON', 1369), ('GPE', 521),
 ('ORGANIZATION', 504), ('PHONE_NUMBER', 350), ('DATE_TIME', 219),
 ('TITLE', 142), ('CREDIT_CARD', 136), ('US_SSN', 80), ('AGE', 74), ('NRP', 55),
 ('ZIP_CODE', 50), ('EMAIL_ADDRESS', 49), ('DOMAIN_NAME', 37),
 ('IP_ADDRESS', 22), ('IBAN_CODE', 21), ('US_DRIVER_LICENSE', 9)]

Min and max number of tokens in dataset: Min: 3, Max: 78
Min and max sentence length in dataset: Min: 9, Max: 407

Example InputSample:
Full text: The address of Persint is 6750 Koskikatu 25 Apt. 864
Artilleros
, CO
 Uruguay 64677
Spans: [Span(type: STREET_ADDRESS, value: 6750 Koskikatu 25 Apt. 864
Artilleros
, CO
 Uruguay 64677, char_span: [26: 83]), Span(type: ORGANIZATION, value: Persint, char_span: [15: 22])]
```

In [6]:
```
print("A few examples sentences containing each entity:\n")
for entity in entity_counts.keys():
    samples = [sample for sample in dataset if entity in set(sample.tags)]
    if len(samples) > 1 and entity != "O":
        print(f"Entity: <{entity}> two example sentences:\n"
              f"\n1) {samples[0].full_text}"
              f"\n2) {samples[1].full_text}"
              f"\n-------------------------------------\n")
```

```
A few examples sentences containing each entity:

Entity: <ORGANIZATION> two example sentences:

1) The address of Persint is 6750 Koskikatu 25 Apt. 864
Artilleros
, CO
 Uruguay 64677
2) The Exversion Orchestra was founded in 1977. Since then, it has grown from a volunteer community orchest
ra to a fully professional orchestra serving Southern Tunisia
------------------------------------

Entity: <STREET_ADDRESS> two example sentences:

1) The address of Persint is 6750 Koskikatu 25 Apt. 864
Artilleros
, CO
 Uruguay 64677
2) Billing address: Sara Schwarz
    28245 Puruntie 82 Apt. 595
   LAPPEENRANTA
    SK
    53650
------------------------------------

Entity: <PERSON> two example sentences:

1) Krisztián Szöllösy listed his top 20 songs for Entertainment Weekly and had the balls to list this song
at #15. (What did he put at #1 you ask? Answer:"Tube Snake Boogie" by Szabina J Gelencsér ɹ€“ go figure)
2) My name is Rubija
------------------------------------

Entity: <DATE_TIME> two example sentences:

1) The Exversion Orchestra was founded in 1977. Since then, it has grown from a volunteer community orchest
ra to a fully professional orchestra serving Southern Tunisia
2) When: 2000-04-16 11:34:35
Where: UPTON SCUDAMORE Country Club.
------------------------------------

Entity: <GPE> two example sentences:
```

```
1) The Exversion Orchestra was founded in 1977. Since then, it has grown from a volunteer community orchest
ra to a fully professional orchestra serving Southern Tunisia
2) I will be travelling to Canada next week, so I need my passport to be ready by then
------------------------------------

Entity: <CREDIT_CARD> two example sentences:

1) What is the limit for card 4454794511390933?
2) My card 4131034282458809939 is expiring this month. Please let me know process to it's extend validity.
------------------------------------

Entity: <US_SSN> two example sentences:

1) Here's my SSN: 460-89-9847
2) Here's my SSN: 514-69-0360
------------------------------------

Entity: <US_DRIVER_LICENSE> two example sentences:

1) my driver's license number is 2270-66-1551
2) My driver's license number is 6940579
------------------------------------

Entity: <AGE> two example sentences:

1) This 79 year old female complaining of stomach pain.
2) This 74 year old female complaining of stomach pain.
------------------------------------

Entity: <ZIP_CODE> two example sentences:

1) Billing address: Sara Schwarz
    28245 Puruntie 82 Apt. 595
   LAPPEENRANTA
    SK
    53650
2) ZIP: 3520
------------------------------------

Entity: <TITLE> two example sentences:

1) My name appears incorrectly on credit card statement could you please correct it to Dr. Cettina Fanucci?
```

```
2) My name appears incorrectly on credit card statement could you please correct it to Dr. Berta Szöllössy?
------------------------------------

Entity: <DOMAIN_NAME> two example sentences:

1) My website is http://www.ScrapbookInsider.com.pt/
2) I've shared files with you https://ClickPhobia.com.br/
------------------------------------

Entity: <EMAIL_ADDRESS> two example sentences:

1) Could you please send me the last billed amount for cc 4007070753690781 on my e-mail UtaKortig@jourrapid
e.com?
2) You said your email is UshurmaDratchev@rhyta.com. Is that correct?
------------------------------------

Entity: <PHONE_NUMBER> two example sentences:

1) I have done an online order but didn't get any message on my registered 905-674-3793. Could you please l
ook into it ?
2) Janka M. Szász

Network and computer systems administrator

Personal Info:
Phone:
60-56-85-91

E-mail:
SzaszJanka@cuvox.de

Website:
https://www.UEarly.se/

Address:
Brucker Bundesstrasse 31 Zezig Streets
 Suite 245
 FÜRLING
 Austria 25173.
------------------------------------

Entity: <IBAN_CODE> two example sentences:
```

```
1) Are there any charges applied for money transfer from GB56HXDO88167774656119 to other bank accounts
2) My IBAN is GB59IFUE40226315499137
----------------------------------

Entity: <IP_ADDRESS> two example sentences:

1) Inject SELECT * FROM Users WHERE client_ip = ?%//!%20\|106.31.73.20|%20/
2) Inject SELECT * FROM Users WHERE client_ip = ?%//!%20\|86.121.97.248|%20/
----------------------------------

Entity: <NRP> two example sentences:

1) From the film English graffiti (also features Kate M Begum. What's not to love?
2) The restaurant is located at 704 1436 Redbud Drive
Cite Essanaouber
, 32
 60810. It serves great Slovenian food.
----------------------------------
```

3. Define the AnalyzerEngine object

Using Presidio with default parameters (not recommended, it's used here for simplicity). For an example on customization, see notebook 5

In [7]:
```
from presidio_analyzer import AnalyzerEngine
# Loading the vanilla Analyzer Engine, with the default NER model.
analyzer_engine = AnalyzerEngine(default_score_threshold=0.4)

pprint(f"Supported entities for English:")
pprint(analyzer_engine.get_supported_entities("en"), compact=True)

print(f"\nLoaded recognizers for English:")
pprint([rec.name for rec in analyzer_engine.registry.get_recognizers("en", all_fields=True)], compact=True

print(f"\nLoaded NER models:")
pprint(analyzer_engine.nlp_engine.models)
```

```
'Supported entities for English:'
['CRYPTO', 'US_DRIVER_LICENSE', 'IP_ADDRESS', 'LOCATION', 'NRP',
 'US_BANK_NUMBER', 'URL', 'US_SSN', 'CREDIT_CARD', 'IBAN_CODE', 'PHONE_NUMBER',
 'PERSON', 'US_ITIN', 'EMAIL_ADDRESS', 'DATE_TIME', 'UK_NHS', 'MEDICAL_LICENSE',
 'US_PASSPORT']

Loaded recognizers for English:
['CreditCardRecognizer', 'UsBankRecognizer', 'UsLicenseRecognizer',
 'UsItinRecognizer', 'UsPassportRecognizer', 'UsSsnRecognizer', 'NhsRecognizer',
 'CryptoRecognizer', 'DateRecognizer', 'EmailRecognizer', 'IbanRecognizer',
 'IpRecognizer', 'MedicalLicenseRecognizer', 'PhoneRecognizer', 'UrlRecognizer',
 'SpacyRecognizer']

Loaded NER models:
[{'lang_code': 'en', 'model_name': 'en_core_web_lg'}]
```

4. Align the dataset's entities to Presidio's entities

There is possibly a difference between the names of entities in the dataset, and the names of entities Presidio can detect. For example, it could be that a dataset labels a name as PER while Presidio returns PERSON. To be able to compare the predicted value to the actual and gather metrics, an alignment between the entity names is necessary. Consider changing the mapping if your dataset and/or Presidio instance supports difference entity types.

In [8]:
```
from presidio_evaluator.models import  PresidioAnalyzerWrapper

entities_mapping=PresidioAnalyzerWrapper.presidio_entities_map # default mapping

print("Using this mapping between the dataset and Presidio's entities:")
pprint(entities_mapping, compact=True)


dataset = SpanEvaluator.align_entity_types(
    dataset,
    entities_mapping=entities_mapping,
    allow_missing_mappings=True
)
new_entity_counts = get_entity_counts(dataset)
print("\nCount per entity after alignment:")
pprint(new_entity_counts.most_common(), compact=True)
```

```
dataset_entities = list(new_entity_counts.values())
```

```
Using this mapping between the dataset and Presidio's entities:
{'ADDRESS': 'LOCATION',
 'AGE': 'AGE',
 'BIRTHDAY': 'DATE_TIME',
 'CITY': 'LOCATION',
 'CREDIT_CARD': 'CREDIT_CARD',
 'CREDIT_CARD_NUMBER': 'CREDIT_CARD',
 'DATE': 'DATE_TIME',
 'DATE_OF_BIRTH': 'DATE_TIME',
 'DATE_TIME': 'DATE_TIME',
 'DOB': 'DATE_TIME',
 'DOMAIN': 'URL',
 'DOMAIN_NAME': 'URL',
 'EMAIL': 'EMAIL_ADDRESS',
 'EMAIL_ADDRESS': 'EMAIL_ADDRESS',
 'FACILITY': 'LOCATION',
 'FIRST_NAME': 'PERSON',
 'GPE': 'LOCATION',
 'HCW': 'PERSON',
 'HOSP': 'ORGANIZATION',
 'HOSPITAL': 'ORGANIZATION',
 'IBAN': 'IBAN_CODE',
 'IBAN_CODE': 'IBAN_CODE',
 'ID': 'ID',
 'IP_ADDRESS': 'IP_ADDRESS',
 'LAST_NAME': 'PERSON',
 'LOC': 'LOCATION',
 'LOCATION': 'LOCATION',
 'NAME': 'PERSON',
 'NATIONALITY': 'NRP',
 'NORP': 'NRP',
 'NRP': 'NRP',
 'O': 'O',
 'ORG': 'ORGANIZATION',
 'ORGANIZATION': 'ORGANIZATION',
 'PATIENT': 'PERSON',
 'PATORG': 'ORGANIZATION',
 'PER': 'PERSON',
 'PERSON': 'PERSON',
 'PHONE': 'PHONE_NUMBER',
 'PHONE_NUMBER': 'PHONE_NUMBER',
 'PREFIX': 'TITLE',
```

```
 'SSN': 'US_SSN',
 'STAFF': 'PERSON',
 'STREET_ADDRESS': 'LOCATION',
 'TIME': 'DATE_TIME',
 'TITLE': 'TITLE',
 'URL': 'URL',
 'US_DRIVER_LICENSE': 'US_DRIVER_LICENSE',
 'US_SSN': 'US_SSN',
 'VENDOR': 'ORGANIZATION',
 'ZIP': 'ZIP_CODE',
 'ZIP_CODE': 'ZIP_CODE'}

Count per entity after alignment:
[('O', 19626), ('LOCATION', 3592), ('PERSON', 1369), ('ORGANIZATION', 504),
 ('PHONE_NUMBER', 350), ('DATE_TIME', 219), ('TITLE', 142),
 ('CREDIT_CARD', 136), ('US_SSN', 80), ('AGE', 74), ('NRP', 55),
 ('ZIP_CODE', 50), ('EMAIL_ADDRESS', 49), ('URL', 37), ('IP_ADDRESS', 22),
 ('IBAN_CODE', 21), ('US_DRIVER_LICENSE', 9)]
```

5. Set up the Evaluator object

In [9]:
```
# Set up the experiment tracker to log the experiment for reproducibility
experiment = get_experiment_tracker()

# Create the evaluator object
evaluator = SpanEvaluator(model=analyzer_engine)


# Track model and dataset params
params = {"dataset_name": dataset_name,
          "model_name": evaluator.model.name}
params.update(evaluator.model.to_log())
experiment.log_parameters(params)
experiment.log_dataset_hash(dataset)
experiment.log_parameter("entity_mappings", json.dumps(entities_mapping))
```

```
--------
Entities supported by this Presidio Analyzer instance:
CRYPTO, US_DRIVER_LICENSE, IP_ADDRESS, LOCATION, NRP, US_BANK_NUMBER, URL, US_SSN, CREDIT_CARD, IBAN_CODE,
PHONE_NUMBER, PERSON, US_ITIN, EMAIL_ADDRESS, DATE_TIME, UK_NHS, MEDICAL_LICENSE, US_PASSPORT
```

```
/presidio_evaluator/evaluation/base_evaluator.py:83: UserWarning: skip words not provided, using default sk
ip words. If you want the evaluation to not use skip words, pass skip_words=[]
  warnings.warn("skip words not provided, using default skip words. "
```

6. Run experiment

In [10]:
```
%%time

## Run experiment

evaluation_results = evaluator.evaluate_all(dataset)
results = evaluator.calculate_score(evaluation_results)

# Track experiment results
experiment.log_metrics(results.to_log())
entities, confmatrix = results.to_confusion_matrix()
experiment.log_confusion_matrix(matrix=confmatrix,
                                labels=entities)

# end experiment
experiment.end()
```

```
Running model PresidioAnalyzerWrapper on dataset...
Finished running model on dataset
saving experiment data to /presidio-research/notebooks/experiment_20250803-230832.json
CPU times: user 9.7 s, sys: 124 ms, total: 9.82 s
Wall time: 10.2 s
```

7. Evaluate results

In [20]:
```
# Plot output
plotter = Plotter(results=results,
                  model_name = evaluator.model.name,
                  save_as="svg",
                  beta = 2)

# Path of the directory to save the plots
output_folder = Path(Path.cwd().parent, "plotter_output")
plotter.plot_scores(output_folder=output_folder)
```

In [12]:
```
pprint({"PII F":results.pii_f, "PII recall": results.pii_recall, "PII precision": results.pii_precision})
```

```
{'PII F': 0.6178033658104517,
 'PII precision': 0.6849427168576104,
 'PII recall': 0.6030259365994236}
```

8. Error analysis

Now let's look into results to understand what's behind the metrics we're getting. Note that evaluation is never perfect. Some things to consider:

1. There's often a mismatch between the annotated span and the predicted span, which isn't necessarily a mistake. For example: `<Southern France>` compared with `Southern <France>`. In the second text, the word `Southern` was not annotated/predicted as part of the entity, but that's not necessarily an error.
2. The synthetic dataset used here isn't representative of a real dataset. Consider using more realistic datasets for evaluation

In [13]:
```
plotter.plot_confusion_matrix(entities=entities, confmatrix=confmatrix, output_folder=output_folder)
```

In [14]:
```
plotter.plot_most_common_tokens(output_folder=output_folder)
```

7a. False positives

Most common false positive tokens:

In [15]:
```
ModelError.most_common_fp_tokens(results.model_errors)
```

```
Most common false positive tokens:
[('8', 15),
 ('sunday labor', 13),
 ('greek', 10),
 ('rościsław dudek dobrosław tomaszewski owens duran kathrine filemonsen dds '
  'yuito hirai abby holloway irena bílá',
  9),
 ('margaret s. stouffer kristen rocher panjiva riku andou jeremy hartmann '
  'ástríður steinarsdóttir impi nummelin',
  9),
 ('60s', 8),
 ('greenlander', 8),
 ('couple', 7),
 ('10th', 7),
 ('cyprus', 7)]
---------------
Example sentence with each FP token:
        - 8 + years (`8` pred as DATE_TIME)
        - the last Sunday before Labor Day (`sunday labor` pred as DATE_TIME)
        - Greek (`greek` pred as O)
        - Rościsław Dudek Dobrosław Tomaszewski Owens Duran Kathrine Filemonsen DDS Yuito Hirai Abby Hollow
ay Irena Bílá (`rościsław dudek dobrosław tomaszewski owens duran kathrine filemonsen dds yuito hirai abby
holloway irena bílá` pred as O)
        - Margaret S. Stouffer Kristen Rocher Panjiva Riku Andou Jeremy Hartmann Ástríður Steinarsdóttir Im
pi Nummelin (`margaret s. stouffer kristen rocher panjiva riku andou jeremy hartmann ástríður steinarsdótti
r impi nummelin` pred as O)
        - the 60s (`60s` pred as DATE_TIME)
        - Greenlander (`greenlander` pred as O)
        - a couple of months (`couple` pred as DATE_TIME)
        - 10th year (`10th` pred as DATE_TIME)
        - Cyprus (`cyprus` pred as O)
```

Out[15]:
```
[('8', 15),
 ('sunday labor', 13),
 ('greek', 10),
 ('rościsław dudek dobrosław tomaszewski owens duran kathrine filemonsen dds yuito hirai abby holloway ire
na bílá',
  9),
 ('margaret s. stouffer kristen rocher panjiva riku andou jeremy hartmann ástríður steinarsdóttir impi num
melin',
  9),
 ('60s', 8),
 ('greenlander', 8),
 ('couple', 7),
 ('10th', 7),
 ('cyprus', 7)]
```

More FP analysis

In [16]:
```
fps_df = ModelError.get_fps_dataframe(results.model_errors, entity=["PERSON"])
fps_df[["full_text", "token", "annotation", "prediction"]].head(20)
```

Out[16]:

	full_text	token	annotation	prediction
0	Answer:"Tube Snake Boogie	answer:"tube snake boogie	O	PERSON
1	nyberg	nyberg	LOCATION	PERSON
2	vitoria	vitoria	LOCATION	PERSON
3	Victoria	victoria	LOCATION	PERSON
4	Csanád	csanád	LOCATION	PERSON
5	Selma	selma	LOCATION	PERSON
6	Sandgreen	sandgreen	LOCATION	PERSON
7	Fletcher Hill	fletcher hill	LOCATION	PERSON
8	Josefsen	josefsen	LOCATION	PERSON
9	Sokolov	sokolov	NRP	PERSON
10	Antonio	antonio	LOCATION	PERSON
11	Florence	florence	LOCATION	PERSON
12	Leigha Mackay	leigha mackay	LOCATION	PERSON
13	Verda	verda	LOCATION	PERSON
14	nyberg	nyberg	LOCATION	PERSON
15	vitoria	vitoria	LOCATION	PERSON
16	Victoria	victoria	LOCATION	PERSON
17	Csanád	csanád	LOCATION	PERSON
18	Selma	selma	LOCATION	PERSON
19	Sandgreen	sandgreen	LOCATION	PERSON

7b. False negatives (FN)

Most common false negative examples + a few samples with FN

In [17]:
```
ModelError.most_common_fn_tokens(results.model_errors, n=15)
```

```
Most common false negative tokens:
[('greenlander', 11),
 ('dr.', 9),
 ('ms.', 9),
 ('64', 7),
 ('46', 6),
 ('61', 6),
 ('47', 6),
 ('65', 6),
 ('54', 5),
 ('21', 4),
 ('78', 4),
 ('63', 4),
 ('35', 4),
 ('40', 4),
 ('22', 4)]
---------------
Example sentence with each FN token:
        - Greenlander (`greenlander` annotated as O)
        - Dr. (`dr.` annotated as O)
        - Ms. (`ms.` annotated as O)
        - 64 (`64` annotated as O)
        - 46 (`46` annotated as O)
        - 61 (`61` annotated as O)
        - 47 (`47` annotated as O)
        - 65 (`65` annotated as O)
        - 54 (`54` annotated as O)
        - 21 (`21` annotated as O)
        - 78 (`78` annotated as O)
        - 63 (`63` annotated as O)
        - 35 (`35` annotated as O)
        - 40 (`40` annotated as O)
        - 22 (`22` annotated as O)
```

Out[17]:
```
[('greenlander', 11),
 ('dr.', 9),
 ('ms.', 9),
 ('64', 7),
 ('46', 6),
 ('61', 6),
 ('47', 6),
 ('65', 6),
 ('54', 5),
 ('21', 4),
 ('78', 4),
 ('63', 4),
 ('35', 4),
 ('40', 4),
 ('22', 4)]
```

More FN analysis

In [18]:
```
fns_df = ModelError.get_fns_dataframe(results.model_errors, entity=["PHONE_NUMBER"])
```

In [19]:
```
fns_df[["full_text", "token", "annotation", "prediction"]].head(20)
```

Out[19]:

	full_text	token	annotation	prediction
0	2270 - 66 - 1551	2270 66 1551	PHONE_NUMBER	US_DRIVER_LICENSE
1	060426070011	060426070011	PHONE_NUMBER	CREDIT_CARD
2	2270 - 66 - 1551	2270 66 1551	PHONE_NUMBER	US_DRIVER_LICENSE
3	060426070011	060426070011	PHONE_NUMBER	CREDIT_CARD