Sentencizer CLASS

</> SOURCE

STRING NAME: `sentencizer` **TRAINABLE:** ⊗

Pipeline component for rule-based sentence boundary detection

A simple pipeline component to allow custom sentence boundary detection logic that doesn't require the dependency parse. By default, sentence segmentation is performed by the `DependencyParser` ≣ , so the `Sentencizer` lets you implement a simpler, rule-based strategy that doesn't require a statistical model to be loaded.

Assigned Attributes

Calculated values will be assigned to `Token.is_sent_start`. The resulting sentences can be accessed using `Doc.sents`.

LOCATION	VALUE
`Token.is_sent_start`	A boolean value indicating whether the token starts a sentence. This will be either `True` or `False` for all tokens. **TYPE:** `bool`
`Doc.sents`	An iterator over sentences in the `Doc`, determined by `Token.is_sent_start` values. **TYPE:** `Iterator[Span]`

Config and implementation

The default config is defined by the pipeline component factory and describes how the component should be configured. You can override its settings via the `config` argument on `nlp.add_pipe` or in your `config.cfg` for training.

SETTING	DESCRIPTION
`punct_chars`	Optional custom list of punctuation characters that mark sentence ends. See below for defaults if not set. Defaults to `None`. **TYPE:** `Optional[List[str]]`
`overwrite` V3.2	Whether existing annotation is overwritten. Defaults to `False`. **TYPE:** `bool`
`scorer` V3.2	The scoring method. Defaults to `Scorer.score_spans` for the attribute `"sents"` **TYPE:** `Optional[Callable]`

```
# cython: infer_types=True, binding=True
from typing import Callable, List, Optional

import srsly

from ..tokens.doc cimport Doc

from .. import util
from ..language import Language
from .pipe import Pipe
from .senter import senter_score

# see #9050
BACKWARD_OVERWRITE = False


@Language.factory(
    "sentencizer",
    assigns=["token.is_sent_start", "doc.sents"],
    default_config={"punct_chars": None, "overwrite": False, "scorer": {"@scorers":
```

Sentencizer.__init__ METHOD

Initialize the sentencizer.

NAME	DESCRIPTION
KEYWORD-ONLY	
`punct_chars`	Optional custom list of punctuation characters that mark sentence ends. See below for defaults. **TYPE:** `Optional[List[str]]`
`overwrite` V3.2	Whether existing annotation is overwritten. Defaults to `False`. **TYPE:** `bool`
`scorer` V3.2	The scoring method. Defaults to `Scorer.score_spans` for the attribute `"sents"` **TYPE:** `Optional[Callable]`

```
['!', '.', '?', '։', '؟', '۔', '܀', '܁', '܂', '߹', '।', '॥', '၊', '။', '።',
 '፧', '፨', '᙮', '᜵', '᜶', '᠃', '᠉', '᥄', '᥅', '᪨', '᪩', '᪪', '᪫',
 '᭚', '᭛', '᭞', '᭟', '᰻', '᰼', '᱾', '᱿', '‼', '‽', '⁇', '⁈', '⁉',
 '⸮', '⸼', '꓿', '꘎', '꘏', '꛳', '꛷', '꡶', '꡷', '꣎', '꣏', '꤯', '꧈',
 '꧉', '꩝', '꩞', '꩟', '꫰', '꫱', '꯫', '﹒', '﹖', '﹗', '！', '．', '？',
 '𐩖', '𐩗', '𑁇', '𑁈', '𑂾', '𑂿', '𑃀', '𑃁', '𑅁', '𑅂', '𑅃', '𑇅',
 '𑇆', '𑇍', '𑇞', '𑇟', '𑈸', '𑈹', '𑈻', '𑈼', '𑊩', '𑑋', '𑑌', '𑗂',
 '𑗃', '𑗉', '𑗊', '𑗋', '𑗌', '𑗍', '𑗎', '𑗏', '𑗐', '𑗑', '𑗒', '𑗓',
 '𑗔', '𑗕', '𑗖', '𑗗', '𑙁', '𑙂', '𑜼', '𑜽', '𑜾', '𑩂', '𑩃', '𑪛',
 '𑪜', '𑱁', '𑱂', '𖩮', '𖩯', '𖫵', '𖬷', '𖬸', '𖭄', '𛲟', '𝪈', '｡', '。']
```

Sentencizer.__call__ METHOD

Apply the sentencizer on a `Doc`. Typically, this happens automatically after the component has been added to the pipeline using `nlp.add_pipe`.

NAME	DESCRIPTION
doc	The Doc object to process, e.g. the Doc in the pipeline. TYPE: Doc
RETURNS	The modified Doc with added sentence boundaries. TYPE: Doc

Sentencizer.pipe METHOD

Apply the pipe to a stream of documents. This usually happens under the hood when the nlp object is called on a text and all pipeline components are applied to the Doc in order.

NAME	DESCRIPTION
stream	A stream of documents. TYPE: Iterable[Doc]
KEYWORD-ONLY	
batch_size	The number of documents to buffer. Defaults to 128 . TYPE: int
YIELDS	The processed documents in order. TYPE: Doc

Sentencizer.to_disk METHOD

Save the sentencizer settings (punctuation characters) to a directory. Will create a file sentencizer.json . This also happens automatically when you save an nlp object with a sentencizer added to its pipeline.

NAME	DESCRIPTION
`path`	A path to a JSON file, which will be created if it doesn't exist. Paths may be either strings or `Path` -like objects. **TYPE:** `Union[str,Path]`

Sentencizer.from_disk METHOD

Load the sentencizer settings from a file. Expects a JSON file. This also happens automatically when you load an `nlp` object or model with a sentencizer added to its pipeline.

NAME	DESCRIPTION
`path`	A path to a JSON file. Paths may be either strings or `Path` -like objects. **TYPE:** `Union[str,Path]`
RETURNS	The modified `Sentencizer` object. **TYPE:** `Sentencizer`

Sentencizer.to_bytes METHOD

Serialize the sentencizer settings to a bytestring.

NAME	DESCRIPTION
RETURNS	The serialized data. **TYPE:** `bytes`

Sentencizer.from_bytes METHOD

Load the pipe from a bytestring. Modifies the object in place and returns it.

NAME	DESCRIPTION
`bytes_data`	The bytestring to load. **TYPE:** `bytes`
RETURNS	The modified `Sentencizer` object. **TYPE:** `Sentencizer`

</> SUGGEST EDITS