Download @ http://nltk.org/

The NLTK FrameNet API

FrameNet website: framenet.icsi.berkeley.edu

Designing for Discoverability with a Rich Linguistic Resource

- **Berkeley FrameNet** is
  - a *lexicon* of >1000 semantic **frames** describing roles and relationships of scenes evoked by English **lexical units** (LUs)
  - a *corpus* of >100k sentences annotated for frames and their roles (arguments)
  - stored in an intricate XML format
- This work: **a convenient Python API for accessing FrameNet** (lexicon + annotated sentences) programmatically and interactively.

Installation

```
>>> import nltk
>>> nltk.download('framenet_v17')
```

Importing

```
>>> from nltk.corpus import framenet as fn
```

Listing frames & LUs by name

```
>>> frames = fn.frames()
>>> frames
[<frame ID=2031 name=Abandonment>, <frame ID=262
name=Abounding_with>, ...]
>>> len(frames)
1221
>>> fn.frames('(?i)creat')
[<frame ID=268 name=Cooking_creation>, <frame ID=1658
name=Create_physical_artwork>, ...]
>>> fn.lus('prepare')
[<lu ID=4894 name=prepare.v>, <lu ID=5507 name=prepare.v>, ...]
>>> fn.lus('prepare', frame='Cooking_creation')
[<lu ID=4894 name=prepare.v>]
```


Nathan Schneider — Georgetown University — http://nathan.cl

Chuck Wooters — Semantic Machines — http://www1.icsi.berkeley.edu/~wooters/

Presented by Michael Roth

EMNLP 2017 • Copenhagen

Frame display

```
>>> f = fn.frame('Cooking_creation')
>>> f
frame (268): Cooking_creation

[URL] https://framenet2.icsi.berkeley.edu/fnReports/data/frame/Cooking_creation.xml

[definition]
  This frame describes food and meal preparation.  A Cook creates a
  Produced_food from (raw) Ingredients.  The Heating_Instrument
  and/or the Container may also be specified.  'Caitlin baked some
  cookies from the pre-packaged dough.'

[semTypes] 0 semantic types

[frameRelations] 3 frame relations
  <Parent=Intentionally_create -- Inheritance -> Child=Cooking_creation>
  <Parent=Apply_heat -- Using -> Child=Cooking_creation>
  <MainEntry=Apply_heat -- See_also -> ReferringEntry=Cooking_creation>

[lexUnit] 19 lexical units
  bake.v (4896), baking.n (17899), concoct.v (4890), cook up.v
  (4891), cook.n (16786), cook.v (4893), cooking.n (17900), fix.v
  (17923), fry.v (17905), frying.n (17907), grill.v (17901),
  grilling.n (17902), make.v (4895), preparation.n (17898),
  prepare.v (4894), put together.v (4897), roast.v (17903),
  roasting.n (17904), whip up.v (4892)

[FE] 12 frame elements
            Core: Cook (2298), Produced_food (2299)
     Peripheral: Container (2302), Degree (2308), Heating_instrument (2300), Ingredients
(2311), Manner (2309), Means (2304), Place (2303), Time (2307)
  Extra-Thematic: Purpose (2305), Recipient (2306)

[FEcoreSets] 0 frame element core sets
>>> f.definition
"This frame describes food and meal preparation.  A Cook creates a Produced_food from (raw)
Ingredients.  The Heating_Instrument and/or the Container may also be specified.  'Caitlin
baked some cookies from the pre-packaged dough.' "
```

Annotated sentence display

```
>>> f = fn.frame('Cooking_creation')
>>> f.lexUnit['bake.v'].exemplars[1]
exemplar sentence (805430):
[sentNo] 0
[aPos] 6483425

[LU] (4896) bake.v in Cooking_creation

[frame] (268) Cooking_creation

[text] + [Target] + [FE]

It is also illegal in Norway for a bakery to bake bread on a
                                 --------    **** ----- -----
                                 Cook             Produ Time

Saturday or Sunday .
------------------

 (Produ=Produced_food)
```

`fn.help()` lists the **main methods**:

frames

- `frame()` to look up a frame by its exact name or ID
- `frames()` to get frames matching a name pattern
- `frames_by_lemma()` to get frames containing an LU matching a name pattern
- `frame_ids_and_names()` to get a mapping from frame IDs to names

FEs

- `fes()` to get frame elements (a.k.a. roles) matching a name pattern, optionally constrained by a frame name pattern

LUs

- `lu()` to look up an LU by its ID
- `lus()` to get lexical units matching a name pattern, optionally constrained by frame
- `lu_ids_and_names()` to get a mapping from LU IDs to names

relations

- `frame_relation_types()` to get the different kinds of frame-to-frame relations (Inheritance, Subframe, Using, etc.).
- `frame_relations()` to get the relation instances, optionally constrained by frame(s) or relation type
- `fe_relations()` to get the frame element pairs belonging to a frame-to-frame relation

semtypes

- `semtypes()` to get the different kinds of semantic types that can be applied to FEs, LUs, and entire frames
- `semtype()` to look up a particular semtype by name, ID, or abbreviation
- `semtype_inherits()` to check whether two semantic types have a subtype-supertype relationship in the semtype hierarchy
- `propagate_semtypes()` to apply inference rules that distribute semtypes over relations between FEs

annotated sentences

- `annotations()` to get annotation sets, in which a token in a sentence is annotated with a lexical unit in a frame, along with its frame elements and their syntactic properties; can be constrained by LU name pattern and limited to lexicographic exemplars or full-text. Sentences of full-text annotation can have multiple annotation sets.
- `sents()` to get annotated sentences illustrating one or more lexical units
- `exemplars()` to get sentences of lexicographic annotation, most of which have just 1 annotation set; can be constrained by LU name pattern, frame, and overt FE(s)
- `doc()` to look up a document of full-text annotation by its ID
- `docs()` to get documents of full-text annotation that match a name pattern
- `docs_metadata()` to get metadata about all full-text documents without loading them
- `ft_sents()` to iterate over sentences of full-text annotation