LLaMaPUn C library

Generated by Doxygen 1.8.7

Sat Aug 9 2014 18:57:55

Contents

1	Mod	lule Inde	ex																			1
	1.1	Module	es												 	 		 		 		1
2	Data	a Struct	ure Index																			3
	2.1	Data S	Structures												 	 		 		 		3
3	File	Index																				5
	3.1	File Lis	st												 	 		 		 		5
4	Mod	lule Doc	cumentatio	on	I																	7
	4.1	DNMII	b												 	 		 				7
	4.2	The no	gram librar	у											 	 		 		 		8
	4.3	Paragr	aph Discri	imi	nati	ion									 	 		 		 		9
	4.4	A Libra	ary for Ster	mn	ninç	g W c	ords								 	 		 		 		10
	4.5	Stopwe	ord Library	y											 	 		 		 		11
	4.6	Unicod	de Normali	ize	r.										 	 		 		 		12
5	Data	a Struct	ure Docur	me	nta	ıtion	1															13
	5.1	dnm_c	hunk Stru	ict l	Ref	eren	се								 	 		 		 		13
	5.2	dnm_it	terator Stru	uct	Re	efere	nce								 	 		 		 		13
	5.3	dnm_s	struct Struc	ct F	Refe	eren	се									 		 		 		14
	5.4	hash_e	element_s	trir	ng S	Struc	ot Re	efer	ence						 	 		 		 		14
6	File	Docum	entation																			15
	6.1	dnmlib	h File Ref	fere	enc	е.									 	 		 		 		15
		6.1.1	Function	ı D	ocu	ımer	ntatio	on							 	 		 		 		16
			6.1.1.1	C	rea	ateDI	NM								 	 		 		 		16
			6.1.1.2	C	nnt	ıltera	ator/	Add	Ann	otati	ion .				 	 		 		 		16
			6.1.1.3	C	nnt	ıltera	atorl	Has	Ann	otati	ion .				 	 		 		 		16
			6.1.1.4	C	mnk	ıltera	atorl	Has	Ann	otati	ionIr	her	ited	. .	 	 		 		 		17
			6.1.1.5	C	nnt	ıltera	atori	Nex	t .						 	 		 		 		17
			6.1.1.6	f	ree	DNN	Л.								 	 		 		 		17
			6117	_	no+F)nm(Chil	dra	altor	ator												17

iv CONTENTS

		6.1.1.8	getDnmlterator	17
6.2	llamap	un_ngram	ns.h File Reference	18
	6.2.1	Function	Documentation	18
		6.2.1.1	llamapun_get_ngrams	18
6.3	llamap	un_para_c	discr.h File Reference	18
6.4	stemm	er.h File F	Reference	18
	6.4.1	Function	Documentation	19
		6.4.1.1	close_stemmer	19
		6.4.1.2	init_stemmer	19
		6.4.1.3	morpha_stem	19
6.5	stopwo	ords.h File	Reference	19
	6.5.1	Function	Documentation	19
		6.5.1.1	free_stopwords	19
		6.5.1.2	is_stopword	19
		6.5.1.3	load_stopwords	20
		6.5.1.4	read_stopwords_from_json	20
6.6	unicod	le_normali	izer.h File Reference	20
	6.6.1	Function	Documentation	20
		6.6.1.1	normalize_unicode	20

Module Index

1.1 Modules

Here	ie	2	liet	Λf	all	modi	عمار

DNMlib	7
The ngram library	
Paragraph Discrimination	ç
A Library for Stemming Words	(
Stopword Library	1
Unicode Normalizer	2

2 **Module Index**

Data Structure Index

2.1 Data Structures

Here are	the	data	structures	with	brief	descriptions
----------	-----	------	------------	------	-------	--------------

dnm_chunk	13
dnm_iterator	13
dnm_struct	14
hash_element_string	14

4 Data Structure Index

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

ib.h	15
include.h	??
apun_ngrams.h	18
apun_ngrams_old.h	??
apun_para_discr.h	18
apun_senna_pos.h	??
apun_utils.h	??
rocessor.h	??
mer.h	18
words.h	19
ode normalizer.h	20

6 File Index

Module Documentation

4.1 DNMlib

Most NLP tools work on plain text. However, the XML structure contains useful information about the structure of a document etc. So one tends to switch back and forth. The purpose of this library is to simplify this switching by providing a DNM (document narrative model) and some tools to iterate over it etc.

8 Module Documentation

4.2 The ngram library

Consists of a single function for finding ngrams so far.

4.3 Paragraph Discrimination

Some tools for experiments concerning paragraph discrimination. Paragraph discrimination refers to the idea that in a mathematical document, paragraphs tend to have certain functions. I.e. they're for example proofs, axioms, definitions, ...

10 Module Documentation

4.4 A Library for Stemming Words

Basically, this library is a small wrapper around the (therefore slightly modified) morpha stemmer from the University of Sussex.

4.5 Stopword Library

4.5	Sto	pword	Library
-----	-----	-------	---------

A small library which provides a function to check whether a word is a regarded a stopword.

12 Module Documentation

4.6 Unicode Normalizer

provides tools for normalizing unicode to ascii. It uses libiconv.

Data Structure Documentation

5.1 dnm_chunk Struct Reference

Data Fields

- char * id
- xmlNode * dom_node
- enum dnm level level
- long offset_parent
- long offset_children_start
- · long offset_children_end
- char ** annotations
- size_t number_of_annotations
- size_t annotations_allocated
- char ** inherited_annotations
- size_t number_of_inherited_annotations
- size_t inherited_annotations_allocated
- size_t offset_start
- size_t offset_end

The documentation for this struct was generated from the following file:

• dnmlib.h

5.2 dnm_iterator Struct Reference

Data Fields

- dnmPtr dnm
- enum dnm_level level
- size_t pos
- size_t start
- size_t end

The documentation for this struct was generated from the following file:

• dnmlib.h

5.3 dnm_struct Struct Reference

Data Fields

- xmlDocPtr document
- char * plaintext
- struct hash_element_string * annotation_handle
- struct dnm_chunk * para_level
- struct dnm_chunk * sent_level
- struct dnm chunk * word_level
- size_t size_para_level
- size_t size_sent_level
- size_t size_word_level
- size_t size_plaintext

The documentation for this struct was generated from the following file:

• dnmlib.h

5.4 hash_element_string Struct Reference

Data Fields

- char * string
- UT_hash_handle hh

The documentation for this struct was generated from the following file:

• dnmlib.h

File Documentation

6.1 dnmlib.h File Reference

```
#include <libxml/tree.h>
#include <uthash.h>
#include <string.h>
```

Data Structures

- · struct hash element string
- struct dnm_struct
- struct dnm_chunk
- struct dnm_iterator

Macros

- #define **DNM_NORMALIZE_MATH** (1 << 0)
- #define **DNM_SKIP_MATH** (1 << 1)
- #define ${
 m DNM_SKIP_CITE}$ (1 << 2)

Typedefs

- typedef struct dnm_struct * dnmPtr
- typedef struct dnm_iterator * dnmIteratorPtr

Enumerations

Functions

- dnmPtr createDNM (xmlDocPtr doc, long parameters)
- void freeDNM (dnmPtr dnm)
- dnmlteratorPtr getDnmlterator (dnmPtr dnm, enum dnm_level level)
- dnmlteratorPtr getDnmChildrenIterator (dnmlteratorPtr it)

16 File Documentation

- · int dnmlteratorNext (dnmlteratorPtr it)
- int dnmlteratorPrevious (dnmlteratorPtr it)
- char * getDnmlteratorContent (dnmlteratorPtr it)
- int dnmlteratorHasAnnotation (dnmlteratorPtr it, const char *annotation)
- int dnmlteratorHasAnnotationInherited (dnmlteratorPtr it, const char *annotation)
- void dnmlteratorAddAnnotation (dnmlteratorPtr it, const char *annotation, int writeIntoDOM, int inheritTo← Children)

6.1.1 Function Documentation

6.1.1.1 dnmPtr createDNM (xmlDocPtr doc, long parameters)

creates a DNM

Example call: createDNM(mydoc, DNM_NORMALIZE_MATH | DNM_SKIP_CITE); Memory has to be freed later using freeDNM

See also

freeDNM

Parameters

doc	a pointer to the DOM
parameters	the parameters

Return values

а	pointer to the new DNM

6.1.1.2 void dnmlteratorAddAnnotation (dnmlteratorPtr it, const char * annotation, int writeIntoDOM, int inheritToChildren

adds an annotation to a chunk. Again: A faster way for repeatedly adding one annotation should be implemented.

Parameters

it	iterator referring to the chunk			
annotation	The string representation of the annotation			
writeIntoDOM	If non-zero: The annotation is also written into the DOM			
inheritToChildren	If non-zero: The annotation is inherited to the child chunks			

6.1.1.3 int dnmlteratorHasAnnotation (dnmlteratorPtr it, const char * annotation)

Checks whether a chunk has a certain annotation. Note that a faster way should be implemented, if you want to repeatedly check for one annotation.

Parameters

it	A pointer to an iterator
annotation	The string representation of the annotation

Return values

6.1 dnmlib.h File Reference 17

1 if the chunk has the annotation, 0 otherwise

6.1.1.4 int dnmlteratorHasAnnotationInherited (dnmlteratorPtr it, const char * annotation)

like dnmlteratorHasAnnotation, just for annotations inherited (i.e. annotations from the parent tags)

See also

dnmlteratorHasAnnotation

6.1.1.5 int dnmlteratorNext (dnmlteratorPtr it)

Make an iterator to point to the next chunk

Parameters

it the iterator that shall be incremented	
---	--

Return values

0 if the iterator points to the last element already, otherwise 1

6.1.1.6 void freeDNM (dnmPtr dnm)

frees the DNM

Parameters

dnm	pointer to the DNM to be freed

6.1.1.7 dnmlteratorPtr getDnmChildrenlterator (dnmlteratorPtr it)

creates an iterator for the children of the current position of an iterator, i.e. over the sentences of a certain paragraph, or over the words of a sentence. The new iterator has to be free'd manually as well

Parameters

it	the iterator to which tells us what to iterate over

Return values

the	iterator for the children

6.1.1.8 dnmlteratorPtr getDnmlterator (dnmPtr dnm, enum dnm_level level)

creates an iterator over the document (uses malloc -> has to be free'd later)

Parameters

dnm	the document to be iterated over
level	the chunks we want to iterate over (DNM_LEVEL_PARA, DNM_LEVEL_SENTENCE, DN←
	M_LEVEL_WORD)

18 File Documentation

Return values

а	pointer to the iterator

6.2 Ilamapun_ngrams.h File Reference

```
#include "jsoninclude.h"
#include <libxml/tree.h>
#include <libxml/parser.h>
#include <libxml/xpath.h>
#include <libxml/xpathInternals.h>
```

Functions

json_object * llamapun_get_ngrams (xmlDocPtr doc)

6.2.1 Function Documentation

6.2.1.1 json_object* llamapun_get_ngrams (xmlDocPtr doc)

creates and returns statistics of the unigrams, bigrams, and trigrams found in a document.

Parameters

	IL DOM
doc	the DOM
ucc	LIC DOW

Return values

```
returns the counts as a JSON object
```

6.3 Ilamapun_para_discr.h File Reference

```
#include "jsoninclude.h"
#include <libxml/tree.h>
```

6.4 stemmer.h File Reference

```
#include <stdio.h>
```

Functions

- void init_stemmer ()
- void close_stemmer ()
- void morpha_stem (const char *input, char **output)

Variables

```
    FILE * morpha_instream
```

- $\bullet \ \ \mathsf{FILE} * \mathbf{morpha_outstream}$
- char * morpha_instream_buff_ptr
- char * morpha outstream buff ptr

6.4.1 Function Documentation

```
6.4.1.1 void close_stemmer ( )
```

closes the stemmer

6.4.1.2 void init_stemmer ()

Initializes the stemmer

6.4.1.3 void morpha_stem (const char * input, char ** output)

stems a sentence

Parameters

input	the input string
output	pointer to the stemmed output

6.5 stopwords.h File Reference

```
#include "jsoninclude.h"
```

Functions

- void read_stopwords_from_json (json_object *)
- void load_stopwords ()
- void free_stopwords ()
- int is_stopword (const char *word)

6.5.1 Function Documentation

```
6.5.1.1 void free_stopwords ( )
```

frees the currently loaded set of stopwords

6.5.1.2 int is_stopword (const char * word)

Checks whether a word is regarded a stopword. Note that this function is case sensitive.

20 File Documentation

Parameters

word	the word to be checked

Return values

returns	1 if the word is a stopword, otherwise 0
---------	--

6.5.1.3 void load_stopwords ()

loads the stopwords from a predefined set of math specialized stopwords

6.5.1.4 void read_stopwords_from_json (json_object *)

loads the stopwords from a JSON array

6.6 unicode_normalizer.h File Reference

Functions

• void normalize_unicode (char *input, char **output)

6.6.1 Function Documentation

6.6.1.1 void normalize_unicode (char * input, char ** output)

Creates a normalized copy of a string

Parameters

input	the input string
output	a pointer to the (normalized) output