Bachelor Thesis Documentation

Release v1.0

Bachelor Thesis

Jul 26, 2016

CONTENTS

1 Bachelorarbeit 3
- 1.1 src package 3
  - 1.1.1 Subpackages 3
    - src.clustering package 3
      - Submodules 3
      - src.clustering.cluster_mappings module 3
      - Module contents 4
    - src.eval package 4
      - Submodules 4
      - src.eval.analogy module 4
      - src.eval.concentration module 4
      - src.eval.concept_groups module 5
      - src.eval.eval_vectors module 5
      - src.eval.word_similarity module 5
      - Module contents 6
    - src.guesser package 6
      - Submodules 6
      - src.guesser.svm_guesser module 6
      - Module contents 7
    - src.mapping package 7
      - Submodules 7
      - src.mapping.map_vectors module 7
      - src.mapping.mapping_operations module 7
      - src.mapping.mapthreading module 8
      - Module contents 9
    - src.prep package 9
      - Subpackages 9
      - Module contents 12
    - src.trans_e package 12
      - Submodules 12
      - src.trans_e.add_inverse_relations module 12
      - src.trans_e.clean_relations module 12
      - src.trans_e.contains_entities module 12
      - src.trans_e.convert_relations module 13
      - src.trans_e.differentiate_datasets module 13
      - src.trans_e.partition_data module 13
      - Module contents 13
  - 1.1.2 Module contents 13

2 Indices and tables 15

Python Module Index **17**

Index **19**

Contents:

CHAPTER

ONE

BACHELORARBEIT

1.1 src package

1.1.1 Subpackages

src.clustering package

Submodules

src.clustering.cluster_mappings module

Cool docstring.

`src.clustering.cluster_mappings.`**`aggregate_cluster`** (*points*, *labels*)

Arranges all clusters in a list, where a sublist with all points at index i corresponds with the custer with label i.

Parameters

- **`points`** (*`list`*) – List of datapoints
- **`labels`** (*`list`*) – List of unique cluster labels

Returns list of lists of datapoints belonging to the i-th cluster

Return type list

`src.clustering.cluster_mappings.`**`alt`** (*func*)

`src.clustering.cluster_mappings.`**`cluster_mappings`** (*vector_inpath*, *do_pca=False*, *target_dim=100*, *indices_inpath=None*, *epsilon=2.625*, *min_s=20*)

`src.clustering.cluster_mappings.`**`get_cluster_size`** (*labels*)

`src.clustering.cluster_mappings.`**`init_argparser`** ( )

`src.clustering.cluster_mappings.`**`load_indices`** (*indices_inpath*)

`src.clustering.cluster_mappings.`**`load_mappings_from_model`** (*mapping_inpath*)

`src.clustering.cluster_mappings.`**`main`** ( )

This is the main method. Duh.

`src.clustering.cluster_mappings.`**`resolve_indices`** (*points*, *labels*, *indices_inpath*, *model*) `src.clustering.cluster_mappings.`**`train_clustering_parameters`**(*vector_inpath*, *iterations=10*)

Module contents

src.eval package

Submodules

src.eval.analogy module

class `src.eval.analogy.`**`AnalogyMasterThread`**(*vector_inpath*, *analogy_path*, *per_section*, *logpath*, *n*)

Bases: `threading.Thread`

`prepare`()

`start_threads`()

class `src.eval.analogy.`**`AnalogyWorkerThread`**(*worker_id*, *model*)

Bases: `threading.Thread`

`find_most_similar_cosmul`(*a*, *a_*, *b*)

`run`()

`src.eval.analogy.`**`analogy_eval`**(*vector_inpath*, *analogy_path*, *per_section=False*, *logpath=None*)

`src.eval.analogy.`**`analogy_eval_parallel`**(*vector_inpath*, *analogy_path*, *per_section=False*, *logpath=None*, *threads=1*)

`src.eval.analogy.`**`output`**(*message*, *logpath=None*)

`src.eval.analogy.`**`read_analogies`**(*analogy_path*, *per_section=False*)

`src.eval.analogy.`**`read_analogies_for_parallel`**(*analogy_path*, *per_section=False*)

`src.eval.analogy.`**`rreplace`**(*s*, *old*, *new*, *occurrence*)

src.eval.concentration module

`src.eval.concentration.`**`alt`**(*func*)

`src.eval.concentration.`**`calculate_concentration`**(*model*, *procs*, *logpath=None*, *vector_inpath=’‘*)

`src.eval.concentration.`**`calculate_concentrations`**(*vectors_inpath*, *procs*, *max_n_vectors*, *logpath=None*)

`src.eval.concentration.`**`calculate_loss_of_precision`**(*vector_inpath*, *procs*, *sizes*, *logpath=None*)

`src.eval.concentration.`**`chunks`**(*l*, *n*)

`src.eval.concentration.`**`chunks2`**(*lst*, *n*)

`src.eval.concentration.`**`init_pool_for_deviances`**(*pool_args*)

`src.eval.concentration.`**`init_pool_for_distances`**(*pool_args*) src.eval.concentration.**load_vectors_from_model**(*vector_inpath*, *max_n=None*, *logpath=None*, *indices=False*)

src.eval.concentration.**load_vectors_from_model_parallel**(*vector_inpath*, *procs*, *logpath=None*)

src.eval.concentration.**output**(*message*, *logpath=None*)

src.eval.concentration.**rreplace**(*s*, *old*, *new*, *occurrence*)

src.eval.concept_groups module

src.eval.concept_groups.**main**()

src.eval.concept_groups.**sampleRelations**(*inpath*, *outpath*, *n*, *sample_size*, *freq_constraint*)

src.eval.concept_groups.**sample_part**(*relation_pairs*, *coin_flip*, *sample_size*)

src.eval.concept_groups.**take_sample_from_list**(*samplelist*, *n*)

src.eval.eval_vectors module

src.eval.eval_vectors.**apply_on_input**(*func*, *sets*, *inpath*, **args*)

src.eval.eval_vectors.**find_nearest_neighbors**(*vector_inpath*, *max*, *wordlist*)

src.eval.eval_vectors.**init_argparser**()

src.eval.eval_vectors.**main**()

src.eval.eval_vectors.**opt_callback**(*option*, *opt*, *value*, *parser*)

src.eval.eval_vectors.**output**(*message*, *logpath=None*)

src.eval.eval_vectors.**plot**(*data*, *max*, *dimensions*, *show_plot=False*, *display_names=False*)

src.eval.eval_vectors.**plot_distance_distribution**(*data*, *max*, *show_plot=False*)

src.eval.eval_vectors.**rreplace**(*s*, *old*, *new*, *occurrence*)

src.eval.word_similarity module

class src.eval.word_similarity.**WordSimMasterThread**(*n*, *vector_inpath*, *wordpair_inpath*, *logpath*, *format*)

Bases: threading.Thread

prepare()

remove_unknowns()

start_threads()

class src.eval.word_similarity.**WordSimWorkerThread**(*worker_id*, *pair_queue*, *model*, *y*)

Bases: threading.Thread

run()

src.eval.word_similarity.**capitalize**(*word*)

src.eval.word_similarity.**evaluate_wordpair_sims**(*x*, *y*, *number_of_pairs*)

src.eval.word_similarity.**output**(*message*, *logpath=None*) `src.eval.word_similarity.`**`parallel_word_sim_eval`** (*vector_inpath*, *wordpair_path*, *logpath*, *format='google'*, *threads=1*)

`src.eval.word_similarity.`**`read_wordpairs`** (*wordpair_path*, *format='google'*)

`src.eval.word_similarity.`**`remove_unknowns`** (*x*, *y*)

`src.eval.word_similarity.`**`rreplace`** (*s*, *old*, *new*, *occurrence*)

`src.eval.word_similarity.`**`word_sim_eval`** (*vector_inpath*, *wordpair_path*, *logpath*, *format='google'*)

Module contents

src.guesser package

Submodules

src.guesser.svm_guesser module

`src.guesser.svm_guesser.`**`convert_data`** (*sets_path*, *tql_inpath*, *vector_inpath*)

`src.guesser.svm_guesser.`**`create_corrupt_triples`** (*grouped_pairs*, *entities*)

`src.guesser.svm_guesser.`**`dump_relation_vectors`** (*relation_vectors*, *outpath*)

`src.guesser.svm_guesser.`**`evaluate`** (*model*, *grouped_test*, *relation_vectors*, *entities*)

`src.guesser.svm_guesser.`**`extract_data_from_uri`** (*uri*)

`src.guesser.svm_guesser.`**`get_rank`** (*target*, *ranks*)

`src.guesser.svm_guesser.`**`init_argparser`** ( )

Initialize all arguments for an ArgumentParser object and return it.

@returns {ArgumentParser} argument parser object

`src.guesser.svm_guesser.`**`load_relation_vectors`** (*inpath*)

`src.guesser.svm_guesser.`**`load_vectors`** (*vector_inpath*)

@param vector_inpath: Path to word2vec model file

`src.guesser.svm_guesser.`**`main`** ( )

`src.guesser.svm_guesser.`**`prepare_training`** (*sets_path*, *vector_inpath*)

`src.guesser.svm_guesser.`**`rank_entities`** (*reference*, *solution*, *model*, *entities*)

`src.guesser.svm_guesser.`**`read_freebase_data`** (*sets_path*)

`src.guesser.svm_guesser.`**`read_freebase_file`** (*fb_inpath*)

`src.guesser.svm_guesser.`**`read_tql_file`** (*tql_inpath*)

`src.guesser.svm_guesser.`**`test_coverage`** (*triples*, *model*)

Test the coverage of a dataset consisting of freebase triples on word2vec word embeddings. For every triple (h, l, t), the entities h and t are taken and used for look up in the word2vec model.

@param triples: list of 3-tuples (freebase triples) @param model: gensim word2vec model

`src.guesser.svm_guesser.`**`train`** (*model*, *grouped_train*, *grouped_corrupted*, *lossf*, *relation_types*, *epochs=1000*, *learning_rate=0.01*, *margin=1*)

`src.guesser.svm_guesser.`**`transform_triples`** (*triples*, *relation_types*, *entities*) src.guesser.svm_guesser. **write_data** (*triples*, *found_entities*, *outpath*)

Module contents

src.mapping package

Submodules

src.mapping.map_vectors module

src.mapping.map_vectors. **alt** (*func*)

src.mapping.map_vectors. **construct_nearest_neighbour_graph** (*vector_inpath*)

src.mapping.map_vectors. **dump_defaultdict** (*ddict*, *outpath*, *pickled=True*)

src.mapping.map_vectors. **dump_vector_defaultdict** (*ddict*, *outpath*, *pickled=True*)

src.mapping.map_vectors. **filter_duplicate_vectors** (*vectors_indir*, *vector_outpath*)

src.mapping.map_vectors. **filter_duplicate_vectors_parallelized** (*vectors_indir*, *vector_outpath*, *procs=1*)

src.mapping.map_vectors. **hash_tuple** (*t*)

src.mapping.map_vectors. **index_vectors** (*vector_inpath*, *vector_outpath*, *indexing_outpath*, *subset*)

src.mapping.map_vectors. **init_argparse** ( )

src.mapping.map_vectors. **init_pool** (*args*)

src.mapping.map_vectors. **load_vectors_from_model** (*vector_inpath*, *max_n=None*, *logpath=None*)

src.mapping.map_vectors. **main** ( )

src.mapping.map_vectors. **output** (*message*, *logpath=None*)

src.mapping.map_vectors. **read_subset** (*subset_inpath*)

src.mapping.map_vectors. **rreplace** (*s*, *old*, *new*, *occurrence*)

src.mapping.mapping_operations module

src.mapping.mapping_operations. **cosine_similarity** (*v1*, *v2*)

src.mapping.mapping_operations. **distance** (*v1*, *v2*)

src.mapping.mapping_operations. **euclidian_distance1** (*v1*, *v2*)

src.mapping.mapping_operations. **euclidian_distance2** (*v1*, *v2*)

src.mapping.mapping_operations. **manhattan_distance** (*v1*, *v2*)

src.mapping.mapping_operations. **soft_cosine_similarity** (*v1*, *v2*)

src.mapping.mapthreading module

class `src.mapping.mapthreading.`**`MappingMasterThread`** (*n*, *vector_inpath*, *vector_outpath*, *features*, *ids_inpath*, *indices_inpath*)

Bases: `threading.Thread`

`prepare` ( )

`read_ids_file` (*ids_inpath*)

`start_threads` ( )

class `src.mapping.mapthreading.`**`MappingWorkerThread`** (*worker_id*, *vector_dict*, *vector_queue*, *vector_outpath*, *features*, *occurrences*, *indices*)

Bases: `threading.Thread`

`concat` (*v1*, *v2*)

`cosine_similarity` (*v1*, *v2*)

`distance` (*v1*, *v2*)

`euclidian_distance1` (*v1*, *v2*)

`euclidian_distance2` (*v1*, *v2*)

`hash_indices` (*i1*, *i2*)

`manhattan_distance` (*v1*, *v2*)

`run` ( )

`spray` (*v1*, *v2*, *cooc*)

class `src.mapping.mapthreading.`**`VectorDict`**

Bases: `object`

`add_skippable` (*index*)

`add_vector` (*index*, *vector*)

`get_keys` ( )

`get_vector` (*index*)

`skippable` (*index_hash*)

`src.mapping.mapthreading.`**`alt`** (*func*)

`src.mapping.mapthreading.`**`init_argparse`** ( )

`src.mapping.mapthreading.`**`main`** ( )

Module contents

src.prep package

Subpackages

src.prep.corpus package

Submodules

src.prep.corpus.convert_to_plain module

`src.prep.corpus.convert_to_plain.`**`alt`** (*func*)

`src.prep.corpus.convert_to_plain.`**`contains_tag`** (*line*)

`src.prep.corpus.convert_to_plain.`**`convert_decow_to_plain`** (*decow_dir*, *out_dir*, *log_path*, *merge_nes*, *log_interval*)

`src.prep.corpus.convert_to_plain.`**`convert_part`** (*argstuple*)

`src.prep.corpus.convert_to_plain.`**`convert_part_merging`** (*argstuple*)

`src.prep.corpus.convert_to_plain.`**`extract_named_entity`** (*line*)

`src.prep.corpus.convert_to_plain.`**`extract_sentence_id`** (*tag*)

`src.prep.corpus.convert_to_plain.`**`get_file_number`** (*filename*)

`src.prep.corpus.convert_to_plain.`**`log_time`** (*logpath='log.txt'*, *interval=5*)

`src.prep.corpus.convert_to_plain.`**`log_time_mp`** (*logpath='log.txt'*, *interval=5*)

`src.prep.corpus.convert_to_plain.`**`main`** ( )

src.prep.corpus.extract_conll module

`src.prep.corpus.extract_conll.`**`extract_conll`** (*inpath*, *outpath*, *column*)

Extract information out of CoNLL files.

Parameters **`inpath`** (*`str`*) – Path to input file.

`src.prep.corpus.extract_conll.`**`init_argparse`** ( )

`src.prep.corpus.extract_conll.`**`main`** ( )

src.prep.corpus.mapper module

src.prep.corpus.prepare_corpus module

`src.prep.corpus.prepare_corpus.`**`construct_yaml_str`** (*self*, *node*)

Docstring for git testing purposes.

`src.prep.corpus.prepare_corpus.`**`get_file_number`** (*filename*)

`src.prep.corpus.prepare_corpus.`**`init_pool`** (*args*) `src.prep.corpus.prepare_corpus.`**`main`** ( )

`src.prep.corpus.prepare_corpus.`**`prepare`** (*ne_inpath*)

`src.prep.corpus.prepare_corpus.`**`process_corpora`** (*nesi*, *corpus_dir*, *out_dir*, *log_dir*, *logging_interval_meta=10*, *logging_interval_processing=10*)

`src.prep.corpus.prepare_corpus.`**`process_corpus`** (*argstuple*)

src.prep.corpus.quick_and_dirty module

`src.prep.corpus.quick_and_dirty.`**`clean_file`** (*inpath*, *outpath*)

src.prep.corpus.reducer module

Module contents

src.prep.misc package

Submodules

src.prep.misc.decorators module

`src.prep.misc.decorators.`**`alt`** (*func*)

`src.prep.misc.decorators.`**`log_time`** (*logpath='log.txt'*, *interval=5*)

`src.prep.misc.decorators.`**`log_time_mp`** (*logpath='log.txt'*, *interval=5*)

Module contents

src.prep.nes package

Submodules

src.prep.nes.extractNE module

`src.prep.nes.extractNE.`**`contains_tag`** (*line*)

`src.prep.nes.extractNE.`**`extract_named_entity`** (*line*)

`src.prep.nes.extractNE.`**`extract_sentence_id`** (*tag*)

`src.prep.nes.extractNE.`**`main`** ( )

`src.prep.nes.extractNE.`**`print_dict_in_file`** (*dictionary*, *out_path*)

`src.prep.nes.extractNE.`**`print_ids_in_file`** (*dictionary*, *out_path*)

`src.prep.nes.extractNE.`**`print_list_in_file`** (*ne_list*, *out_path*)

`src.prep.nes.extractNE.`**`process`** (*inpath*, *outpath*, *logpath*)

src.prep.nes.merge module

src.prep.nes.merge.**dump_ids_dict** (*idsdict*, *outpath*)

src.prep.nes.merge.**freqWorker** (*inpath*)

src.prep.nes.merge.**idWorker** (*inpath*)

src.prep.nes.merge.**load_ids_dict** (*inpath*)

src.prep.nes.merge.**main** ()

src.prep.nes.merge.**mergeDicts** (*dicttuple*)

src.prep.nes.merge.**merge_frequency_files** (*infiles_path*, *outpath*, *logpath*)

src.prep.nes.merge.**merge_id_dicts** (*dicttuple*)

src.prep.nes.merge.**merge_id_files** (*infiles_path*, *outpath*, *logpath*, *yaml=False*)

src.prep.nes.merge.**print_key_lengths** (*dictionary*)

src.prep.nes.merge.**rl** (*infile*)

src.prep.nes.mwe module

src.prep.nes.mwe.**create_mwe_pickle** (*inpath*, *outpath*, *logpath='./mwes.log'*)

src.prep.nes.mwe.**create_mwe_pickle2** (*inpath*, *outpath*, *logpath='./mwes.log'*)

src.prep.nes.mwe.**dump_dict_pickle** (*d*, *outpath*)

src.prep.nes.mwe.**dump_dict_pickle2** (*d*, *outpath*)

src.prep.nes.mwe.**load_dict_pickle** (*inpath*)

src.prep.nes.mwe.**load_dict_pickle2** (*inpath*)

src.prep.nes.mwe.**main** ()

src.prep.nes.mwe.**replace_mwes** (*mwe_path*, *corpus_path*, *out_path*)

src.prep.nes.statistics module

src.prep.nes.statistics.**calculate_occurrences** (*freqpath*, *relations_path*)

src.prep.nes.statistics.**main** ()

Module contents

src.prep.relations package

Submodules

src.prep.relations.relations module

exception src.prep.relations.relations.**MissingTranslationException**
Bases: exceptions.Exception

get_id ( )

src.prep.relations.relations.**fetch_name** (*id*, *lang='en'*)

src.prep.relations.relations.**fetch_relation_triples_of_file** (*inpath*, *outpath*, *logpath*, *lang='en'*)

src.prep.relations.relations.**format_fbid** (*id*)

src.prep.relations.relations.**freebase_request** (*query*, *api_key*, *service_url*)

src.prep.relations.relations.**main** ( )

src.prep.relations.relations.**read_credentials** ( )

src.prep.relations.relations.**rl** (*infile*)

src.prep.relations.relations.**translate_name** (*name*, *lang='en'*)

src.prep.relations.relations.**translate_word2vec_question_phrases** (*inpath*, *outpath*, *lang='en'*)

Module contents

Module contents

src.trans_e package

Submodules

src.trans_e.add_inverse_relations module

src.trans_e.add_inverse_relations.**add_inverse_relations** (*relations_inpath*, *relations_outpath*, *inverse_relations*, *known_relations*)

src.trans_e.add_inverse_relations.**init_argparse** ( )

src.trans_e.add_inverse_relations.**main** ( )

src.trans_e.add_inverse_relations.**read_file_with_inverse_relations** (*inverse_inpath*)

src.trans_e.clean_relations module

src.trans_e.contains_entities module

src.trans_e.contains_entities.**contains_entities** (*entities1*, *entities2*)

src.trans_e.contains_entities.**create_new_dataset** (*entities1*, *dataset*, *outpath*)

src.trans_e.contains_entities.**extract_entities_from_relation_dataset** (*dataset_inpath*)

src.trans_e.contains_entities.**extract_entities_from_tql_file** (*tql_path*)

src.trans_e.contains_entities.**format_fbid** (*id*)

src.trans_e.contains_entities.**init_argparse** ( )

`src.trans_e.contains_entities.`**`main`** ( )

src.trans_e.convert_relations module

src.trans_e.differentiate_datasets module

`src.trans_e.differentiate_datasets.`**`compare_entities`** (*set1*, *set2*)

`src.trans_e.differentiate_datasets.`**`init_argparse`** ( )

`src.trans_e.differentiate_datasets.`**`main`** ( )

`src.trans_e.differentiate_datasets.`**`read_dataset`** (*inpath*)

src.trans_e.partition_data module

`src.trans_e.partition_data.`**`check_data_integrity`** (*data_inpath*, *remove_clones*, *outpath*)

Check whether all triplets in the data are unique.

`src.trans_e.partition_data.`**`check_set_integrity`** (*indir*)

`src.trans_e.partition_data.`**`get_stats`** (*data*)

`src.trans_e.partition_data.`**`init_argparse`** ( )

`src.trans_e.partition_data.`**`main`** ( )

`src.trans_e.partition_data.`**`partition_data`** (*data*, *prts*, *outdir*, *whole=True*)

`src.trans_e.partition_data.`**`partition_relation_wise`** (*data*, *prts*)

`src.trans_e.partition_data.`**`partition_whole`** (*data*, *prts*)

`src.trans_e.partition_data.`**`partitions_list`** (*l*, *prts*)

`src.trans_e.partition_data.`**`read_only_relations_into_set`** (*inpath*)

`src.trans_e.partition_data.`**`read_relations`** (*inpath*)

`src.trans_e.partition_data.`**`write_data_in_file`** (*data*, *outfile*)

Module contents

1.1.2 Module contents

CHAPTER TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

S

src, 13
src.clustering, 4
src.clustering.cluster_mappings, 3
src.eval, 6
src.eval.analogy, 4
src.eval.concentration, 4
src.eval.concept_groups, 5
src.eval.eval_vectors, 5
src.eval.word_similarity, 5
src.guesser, 7
src.guesser.svm_guesser, 6
src.mapping, 9
src.mapping.map_vectors, 7
src.mapping.mapping_operations, 7
src.mapping.mapthreading, 8
src.prep, 12
src.prep.corpus, 10
src.prep.corpus.convert_to_plain, 9
src.prep.corpus.extract_conll, 9
src.prep.corpus.mapper, 9
src.prep.corpus.prepare_corpus, 9
src.prep.corpus.quick_and_dirty, 10
src.prep.corpus.reducer, 10
src.prep.misc, 10
src.prep.misc.decorators, 10
src.prep.nes, 11
src.prep.nes.extractNE, 10
src.prep.nes.merge, 11
src.prep.nes.mwe, 11
src.prep.nes.statistics, 11
src.prep.relations, 12
src.prep.relations.relations, 11
src.trans_e, 13
src.trans_e.add_inverse_relations, 12
src.trans_e.clean_relations, 12
src.trans_e.contains_entities, 12
src.trans_e.convert_relations, 13
src.trans_e.differentiate_datasets, 13
src.trans_e.partition_data, 13

A

add_inverse_relations() (in module src.trans_e.add_inverse_relations), 12
add_skippable() (src.mapping.mapthreading.VectorDict method), 8
add_vector() (src.mapping.mapthreading.VectorDict method), 8
aggregate_cluster() (in module src.clustering.cluster_mappings), 3
alt() (in module src.clustering.cluster_mappings), 3
alt() (in module src.eval.concentration), 4
alt() (in module src.mapping.map_vectors), 7
alt() (in module src.mapping.mapthreading), 8
alt() (in module src.prep.corpus.convert_to_plain), 9
alt() (in module src.prep.misc.decorators), 10
analogy_eval() (in module src.eval.analogy), 4
analogy_eval_parallel() (in module src.eval.analogy), 4
AnalogyMasterThread (class in src.eval.analogy), 4
AnalogyWorkerThread (class in src.eval.analogy), 4
apply_on_input() (in module src.eval.eval_vectors), 5

C

calculate_concentration() (in module src.eval.concentration), 4
calculate_concentrations() (in module src.eval.concentration), 4
calculate_loss_of_precision() (in module src.eval.concentration), 4
calculate_occurrences() (in module src.prep.nes.statistics), 11
capitalize() (in module src.eval.word_similarity), 5
check_data_integrity() (in module src.trans_e.partition_data), 13
check_set_integrity() (in module src.trans_e.partition_data), 13
chunks() (in module src.eval.concentration), 4
chunks2() (in module src.eval.concentration), 4
clean_file() (in module src.prep.corpus.quick_and_dirty), 10
cluster_mappings() (in module src.clustering.cluster_mappings), 3
compare_entities() (in module src.trans_e.differentiate_datasets), 13
concat() (src.mapping.mapthreading.MappingWorkerThread method), 8
construct_nearest_neighbour_graph() (in module src.mapping.map_vectors), 7
construct_yaml_str() (in module src.prep.corpus.prepare_corpus), 9
contains_entities() (in module src.trans_e.contains_entities), 12
contains_tag() (in module src.prep.corpus.convert_to_plain), 9
contains_tag() (in module src.prep.nes.extractNE), 10
convert_data() (in module src.guesser.svm_guesser), 6
convert_decow_to_plain() (in module src.prep.corpus.convert_to_plain), 9
convert_part() (in module src.prep.corpus.convert_to_plain), 9
convert_part_merging() (in module src.prep.corpus.convert_to_plain), 9
cosine_similarity() (in module src.mapping.mapping_operations), 7
cosine_similarity() (src.mapping.mapthreading.MappingWorkerThread method), 8
create_corrupt_triples() (in module src.guesser.svm_guesser), 6
create_mwe_pickle() (in module src.prep.nes.mwe), 11
create_mwe_pickle2() (in module src.prep.nes.mwe), 11
create_new_dataset() (in module src.trans_e.contains_entities), 12

D

distance() (in module src.mapping.mapping_operations), 7
distance() (src.mapping.mapthreading.MappingWorkerThread method), 8
dump_defaultdict() (in module src.mapping.map_vectors), 7
dump_dict_pickle() (in module src.prep.nes.mwe), 11
dump_dict_pickle2() (in module src.prep.nes.mwe), 11
dump_ids_dict() (in module src.prep.nes.merge), 11
dump_relation_vectors() (in module src.guesser.svm_guesser), 6

dump_vector_defaultdict() (in module src.mapping.map_vectors), 7

E

euclidian_distance1() (in module src.mapping.mapping_operations), 7
euclidian_distance1() (src.mapping.mapthreading.MappingWorkerThread method), 8
euclidian_distance2() (in module src.mapping.mapping_operations), 7
euclidian_distance2() (src.mapping.mapthreading.MappingWorkerThread method), 8
evaluate() (in module src.guesser.svm_guesser), 6
evaluate_wordpair_sims() (in module src.eval.word_similarity), 5
extract_conll() (in module src.prep.corpus.extract_conll), 9
extract_data_from_uri() (in module src.guesser.svm_guesser), 6
extract_entities_from_relation_dataset() (in module src.trans_e.contains_entities), 12
extract_entities_from_tql_file() (in module src.trans_e.contains_entities), 12
extract_named_entity() (in module src.prep.corpus.convert_to_plain), 9
extract_named_entity() (in module src.prep.nes.extractNE), 10
extract_sentence_id() (in module src.prep.corpus.convert_to_plain), 9
extract_sentence_id() (in module src.prep.nes.extractNE), 10

F

fetch_name() (in module src.prep.relations.relations), 12
fetch_relation_triples_of_file() (in module src.prep.relations.relations), 12
filter_duplicate_vectors() (in module src.mapping.map_vectors), 7
filter_duplicate_vectors_parallelized() (in module src.mapping.map_vectors), 7
find_most_similar_cosmul() (src.eval.analogy.AnalogyWorkerThread method), 4
find_nearest_neighbors() (in module src.eval.eval_vectors), 5
format_fbid() (in module src.prep.relations.relations), 12
format_fbid() (in module src.trans_e.contains_entities), 12
freebase_request() (in module src.prep.relations.relations), 12
freqWorker() (in module src.prep.nes.merge), 11

G

get_cluster_size() (in module src.clustering.cluster_mappings), 3
get_file_number() (in module src.prep.corpus.convert_to_plain), 9
get_file_number() (in module src.prep.corpus.prepare_corpus), 9
get_id() (src.prep.relations.relations.MissingTranslationException method), 11
get_keys() (src.mapping.mapthreading.VectorDict method), 8
get_rank() (in module src.guesser.svm_guesser), 6
get_stats() (in module src.trans_e.partition_data), 13
get_vector() (src.mapping.mapthreading.VectorDict method), 8

H

hash_indices() (src.mapping.mapthreading.MappingWorkerThread method), 8
hash_tuple() (in module src.mapping.map_vectors), 7

I

idWorker() (in module src.prep.nes.merge), 11
index_vectors() (in module src.mapping.map_vectors), 7
init_argparse() (in module src.mapping.map_vectors), 7
init_argparse() (in module src.mapping.mapthreading), 8
init_argparse() (in module src.prep.corpus.extract_conll), 9
init_argparse() (in module src.trans_e.add_inverse_relations), 12
init_argparse() (in module src.trans_e.contains_entities), 12
init_argparse() (in module src.trans_e.differentiate_datasets), 13
init_argparse() (in module src.trans_e.partition_data), 13
init_argparser() (in module src.clustering.cluster_mappings), 3
init_argparser() (in module src.eval.eval_vectors), 5
init_argparser() (in module src.guesser.svm_guesser), 6
init_pool() (in module src.mapping.map_vectors), 7
init_pool() (in module src.prep.corpus.prepare_corpus), 9
init_pool_for_deviances() (in module src.eval.concentration), 4
init_pool_for_distances() (in module src.eval.concentration), 4

L

load_dict_pickle() (in module src.prep.nes.mwe), 11
load_dict_pickle2() (in module src.prep.nes.mwe), 11
load_ids_dict() (in module src.prep.nes.merge), 11
load_indices() (in module src.clustering.cluster_mappings), 3
load_mappings_from_model() (in module src.clustering.cluster_mappings), 3

load_relation_vectors() (in module src.guesser.svm_guesser), 6
load_vectors() (in module src.guesser.svm_guesser), 6
load_vectors_from_model() (in module src.eval.concentration), 4
load_vectors_from_model() (in module src.mapping.map_vectors), 7
load_vectors_from_model_parallel() (in module src.eval.concentration), 5
log_time() (in module src.prep.corpus.convert_to_plain), 9
log_time() (in module src.prep.misc.decorators), 10
log_time_mp() (in module src.prep.corpus.convert_to_plain), 9
log_time_mp() (in module src.prep.misc.decorators), 10

M

main() (in module src.clustering.cluster_mappings), 3
main() (in module src.eval.concept_groups), 5
main() (in module src.eval.eval_vectors), 5
main() (in module src.guesser.svm_guesser), 6
main() (in module src.mapping.map_vectors), 7
main() (in module src.mapping.mapthreading), 8
main() (in module src.prep.corpus.convert_to_plain), 9
main() (in module src.prep.corpus.extract_conll), 9
main() (in module src.prep.corpus.prepare_corpus), 9
main() (in module src.prep.nes.extractNE), 10
main() (in module src.prep.nes.merge), 11
main() (in module src.prep.nes.mwe), 11
main() (in module src.prep.nes.statistics), 11
main() (in module src.prep.relations.relations), 12
main() (in module src.trans_e.add_inverse_relations), 12
main() (in module src.trans_e.contains_entities), 13
main() (in module src.trans_e.differentiate_datasets), 13
main() (in module src.trans_e.partition_data), 13
manhattan_distance() (in module src.mapping.mapping_operations), 7
manhattan_distance() (src.mapping.mapthreading.MappingWorkerThread method), 8
MappingMasterThread (class in src.mapping.mapthreading), 8
MappingWorkerThread (class in src.mapping.mapthreading), 8
merge_frequency_files() (in module src.prep.nes.merge), 11
merge_id_dicts() (in module src.prep.nes.merge), 11
merge_id_files() (in module src.prep.nes.merge), 11
mergeDicts() (in module src.prep.nes.merge), 11
MissingTranslationException, 11

O

opt_callback() (in module src.eval.eval_vectors), 5
output() (in module src.eval.analogy), 4
output() (in module src.eval.concentration), 5
output() (in module src.eval.eval_vectors), 5
output() (in module src.eval.word_similarity), 5
output() (in module src.mapping.map_vectors), 7

P

parallel_word_sim_eval() (in module src.eval.word_similarity), 5
partition_data() (in module src.trans_e.partition_data), 13
partition_relation_wise() (in module src.trans_e.partition_data), 13
partition_whole() (in module src.trans_e.partition_data), 13
partitions_list() (in module src.trans_e.partition_data), 13
plot() (in module src.eval.eval_vectors), 5
plot_distance_distribution() (in module src.eval.eval_vectors), 5
prepare() (in module src.prep.corpus.prepare_corpus), 10
prepare() (src.eval.analogy.AnalogyMasterThread method), 4
prepare() (src.eval.word_similarity.WordSimMasterThread method), 5
prepare() (src.mapping.mapthreading.MappingMasterThread method), 8
prepare_training() (in module src.guesser.svm_guesser), 6
print_dict_in_file() (in module src.prep.nes.extractNE), 10
print_ids_in_file() (in module src.prep.nes.extractNE), 10
print_key_lengths() (in module src.prep.nes.merge), 11
print_list_in_file() (in module src.prep.nes.extractNE), 10
process() (in module src.prep.nes.extractNE), 10
process_corpora() (in module src.prep.corpus.prepare_corpus), 10
process_corpus() (in module src.prep.corpus.prepare_corpus), 10

R

rank_entities() (in module src.guesser.svm_guesser), 6
read_analogies() (in module src.eval.analogy), 4
read_analogies_for_parallel() (in module src.eval.analogy), 4
read_credentials() (in module src.prep.relations.relations), 12
read_dataset() (in module src.trans_e.differentiate_datasets), 13
read_file_with_inverse_relations() (in module src.trans_e.add_inverse_relations), 12
read_freebase_data() (in module src.guesser.svm_guesser), 6
read_freebase_file() (in module src.guesser.svm_guesser), 6
read_ids_file() (src.mapping.mapthreading.MappingMasterThread method), 8

read_only_relations_into_set() (in module src.trans_e.partition_data), 13
read_relations() (in module src.trans_e.partition_data), 13
read_subset() (in module src.mapping.map_vectors), 7
read_tql_file() (in module src.guesser.svm_guesser), 6
read_wordpairs() (in module src.eval.word_similarity), 6
remove_unknowns() (in module src.eval.word_similarity), 6
remove_unknowns() (src.eval.word_similarity.WordSimMasterThread method), 5
replace_mwes() (in module src.prep.nes.mwe), 11
resolve_indices() (in module src.clustering.cluster_mappings), 3
rl() (in module src.prep.nes.merge), 11
rl() (in module src.prep.relations.relations), 12
rreplace() (in module src.eval.analogy), 4
rreplace() (in module src.eval.concentration), 5
rreplace() (in module src.eval.eval_vectors), 5
rreplace() (in module src.eval.word_similarity), 6
rreplace() (in module src.mapping.map_vectors), 7
run() (src.eval.analogy.AnalogyWorkerThread method), 4
run() (src.eval.word_similarity.WordSimWorkerThread method), 5
run() (src.mapping.mapthreading.MappingWorkerThread method), 8

S

sample_part() (in module src.eval.concept_groups), 5
sampleRelations() (in module src.eval.concept_groups), 5
skippable() (src.mapping.mapthreading.VectorDict method), 8
soft_cosine_similarity() (in module src.mapping.mapping_operations), 7
spray() (src.mapping.mapthreading.MappingWorkerThread method), 8
src (module), 13
src.clustering (module), 4
src.clustering.cluster_mappings (module), 3
src.eval (module), 6
src.eval.analogy (module), 4
src.eval.concentration (module), 4
src.eval.concept_groups (module), 5
src.eval.eval_vectors (module), 5
src.eval.word_similarity (module), 5
src.guesser (module), 7
src.guesser.svm_guesser (module), 6
src.mapping (module), 9
src.mapping.map_vectors (module), 7
src.mapping.mapping_operations (module), 7
src.mapping.mapthreading (module), 8
src.prep (module), 12
src.prep.corpus (module), 10
src.prep.corpus.convert_to_plain (module), 9
src.prep.corpus.extract_conll (module), 9
src.prep.corpus.mapper (module), 9
src.prep.corpus.prepare_corpus (module), 9
src.prep.corpus.quick_and_dirty (module), 10
src.prep.corpus.reducer (module), 10
src.prep.misc (module), 10
src.prep.misc.decorators (module), 10
src.prep.nes (module), 11
src.prep.nes.extractNE (module), 10
src.prep.nes.merge (module), 11
src.prep.nes.mwe (module), 11
src.prep.nes.statistics (module), 11
src.prep.relations (module), 12
src.prep.relations.relations (module), 11
src.trans_e (module), 13
src.trans_e.add_inverse_relations (module), 12
src.trans_e.clean_relations (module), 12
src.trans_e.contains_entities (module), 12
src.trans_e.convert_relations (module), 13
src.trans_e.differentiate_datasets (module), 13
src.trans_e.partition_data (module), 13
start_threads() (src.eval.analogy.AnalogyMasterThread method), 4
start_threads() (src.eval.word_similarity.WordSimMasterThread method), 5
start_threads() (src.mapping.mapthreading.MappingMasterThread method), 8

T

take_sample_from_list() (in module src.eval.concept_groups), 5
test_coverage() (in module src.guesser.svm_guesser), 6
train() (in module src.guesser.svm_guesser), 6
train_clustering_parameters() (in module src.clustering.cluster_mappings), 3
transform_triples() (in module src.guesser.svm_guesser), 6
translate_name() (in module src.prep.relations.relations), 12
translate_word2vec_question_phrases() (in module src.prep.relations.relations), 12

V

VectorDict (class in src.mapping.mapthreading), 8

W

word_sim_eval() (in module src.eval.word_similarity), 6
WordSimMasterThread (class in src.eval.word_similarity), 5
WordSimWorkerThread (class in src.eval.word_similarity), 5
write_data() (in module src.guesser.svm_guesser), 6
write_data_in_file() (in module src.trans_e.partition_data), 13