MATCHER

This module runs a specific matcher (retrieval system) for a given query to produce a ranked list of documents

Input

- Query - id of the query to be run
- Matcher - type of the matcher to use
- Index - path to the index where the preprocessed document collections exists

Output

Ranked list of documents in three file formats:

1) Standard TREC format (.trec)

```
query18062 Q0    MATERIAL_OP2-3B_95836094   1     -2.529     bert-maxp
query18062 Q0    MATERIAL_OP2-3B_49481014   2     -2.556     bert-maxp
…..
```

2) TSV format (.tsv)

```
query18062 toothbrush:
MATERIAL_OP2-3B_95836094   1.00000
MATERIAL_OP2-3B_49481014   1.00000
…..
```

3) ETSV format (.etsv)

```
query18062 toothbrush:
MATERIAL_OP2-3B_95836094   -2.529 3B/IARPA_MATERIAL_OP2-3B/EVAL/text/src
MATERIAL_OP2-3B_49481014   -2.556 3B/IARPA_MATERIAL_OP2-3B/EVAL/text/src
…..
```

Docker Commands

To run the docker images pass the following environment variables to docker using the ‘-e’ flag:

QUERY=<query id>	filename of the pre-processed query from the output of the query analyzer
TYPE=<matcher type>	type of matcher (retrieval system) to use
INDEX=<path to index>	path to the index containing preprocessed data. Multiple paths are divided by '+'
CUTOFF=<cutoff value>	cutoff for the ranked list of documents (default: 1000)
LANG=<language code>	language ISO 639-1 code (tl, sw, kk, etc.)
CONFIG=<config name>	filename of the experiment configuration file
NAME=<matcher name>	name of the matcher used in the experiment configuration file

and volume mounts using '-v' flag:

-v <root_dir>:/media/index	directory to the index root
-v <query_dir>:/media/queries/	directory containing the preprocessed queries
-v <config dir>:/media/input/configs	directory that contains the config file
-v <log_dir>:/media/log_dir/	directory to save logs

Examples

```
docker run --rm \
-e "QUERY=query18016" \
-e "TYPE=bbn_text_fast" \
-e
"INDEX=/storage/data/NIST-data/3S/IARPA_MATERIAL_OP2-3S/EVAL/text/ind
ex_store/indexing-umd-v7.3/indexing_params_v1.0/text/src/indri_T_N" \
-e "CUTOFF=-1" \
-e "LANG=ka" \
-e "CONFIG=hmm.json" \
-e "NAME=hmm" \
```

```
-v /storage/data/NIST-data/:/media/index \
-v
/storage/data/NIST-data/3S/IARPA_MATERIAL_OP2-3S/query_store/query-an
alyzer-umd-v14.2/QUERY2/:/media/queries \
-v /storage/proj/srnair/matcher_development/ka_configs:/media/input/configs/ \
-v /storage/proj/srnair/matcher_development/ka_log:/media/log_dir/ \
--name matching-umd-instance matching-umd:v15.4
```

System Requirements

- CPU
- RAM

Standalone

Yes

Approach

This component supports several different matchers that could be run on all types of queries

- Probabilistic Structured Queries (PSQ) based HMM Model
- Neural Lexical Translation Model
- n-best NMT ranking Model
- Indri Language Model
- Anserini BM25
- Keyword Spotting

Additionally, there exists matchers that support custom types of queries

- Sequential Dependence Model (SDM) for phrase-based queries
- Expansion Model for conceptual queries