D5.4 Robotic Spatial Commands

Project Acronym:	**MARIO**
Project Title:	**Managing active and healthy aging with use of caring service robots**
Project Number:	**643808**
Call:	**H2020-PHC-2014-single-stage**
Topic:	**PHC-19-2014**
Type of Action:	**RIA**

D5.4

Work Package:	*WP5*	
Due Date:		M22
Submission Date:		30/11/2016
Start Date of Project:		01/02/2015
Duration of Project:		36 Months
Organisation Responsible of Deliverable:		PASSAU
Version:		1.0
Status:		Final
Author name(s):	Andre Freitas	University of Passau
Reviewer(s):	Alessandro Russo	Massimiliano Raciti
Nature:	☐ R – Report ☐ P – Prototype ☐ D – Demonstrator ☒ O - Other	
Dissemination level:	☒ PU - Public ☐ CO - Confidential, only for members of the consortium (including the Commission) ☐ RE - Restricted to a group specified by the consortium (including the Commission Services)	

Project co-funded by the European Commission within the Horizon 2020 Programme (2014-2020)

Revision history

Version	Date	Modified by	Comments
V0.1	12 11 16	Andre Freitas	First document draft created.
V0.2	28 11 16	Alessandro Russo Massimiliano Raciti	Reviewers comments received.
V1.0	30 11 16	Andre Freitas	Reviewer's comments addressed.

Executive Summary (1 Page Max)

<u>The executive summary of each deliverable must follow the below format</u>

Subject matter

This report describes the methodology and the main components of the Semantic Correction Component (SCC), which is a task redefinition of the original task 5.4. (Robotic Spatial Commands), taking into account the high priority of having a speech recognition component for MARIO.

How does the deliverable fit into MARIO

The SCC aims at automatically correcting the output of the Automatic Speech Recognition (ASR) component using the combination of semantic and phonetic approximation techniques over the MARIO Knowledge Base (KB) and command frames.

Methods

- *A machine learning-based model based on distributional semantic models was developed to detect which elements in the input are likely to be correct and easier to align with MARIO KB (semantic pivoting method).*

- *Experiments were performed to evaluate the performance of the method over large KBs, under an open domain scenario.*

- *A software component was produced.*

Findings

- *The SCC provides an efficient correction strategy under an initial experimental setting.*

Conclusions

- The semantic pivoting method showed high accuracy over the evaluation scenario.

Table of Contents

Executive Summary (1 Page Max) **5**

Introduction **7**

Purpose and Target of the Deliverable 7

Task Objectives 7

Target Group 7

Relations to other Activities in the Project 7

Document Outline 8

Introduction **8**

Semantic Correction Component (SCC) **8**

Model Description **9**

Problem Description 9

Distributional-Phonological Semantic Approximation Model 9

High-level Components 10

Semantic Pivoting Model 10

Phonetic Representation 11

Distributional Vector Representation 11

Learning to Pivot 11

Slot Filling 12

Experiments & Evaluation 13

Repository 14

Usage 15

Dependencies **16**

References **16**

1. Introduction

1.1. Purpose and Target of the Deliverable

Purpose: This deliverable focuses on the description of the method behind the creation of the semantic correction component, which is a task redefinition from the original Task 5.4.

Outcomes: The key outcome is a semantic matching methodology and system based on a heuristic method for detecting correctly and incorrectly transcribed terms in the input utterances in relation to the MARIO Knowledge Base.

1.2. Task Objectives

This task has the goal of defining a software component which improves the input of the Automatic Speech Recognition Module.

1.3. Target Group

This deliverable is a component of the software infrastructure provided to the MARIO project, targeting to support a natural language interaction with users.

1.4. Relations to other Activities in the Project

This deliverable connects to the following components within the MARIO high-level software architecture:

- **Robot Reading and Listening Component (ASR):** Uses as an input the output of the ASR component (Robot Reading and Listening component investigated in Task 5.2 and presented in D5.2).

- **MARIO Knowledge Base and Ontology Networks:** The component depends on indexing the datasets, operational frames and schemas behind MARIO (Task 5.1, D5.1).

This work takes into account the requirements and architecture defined in WP1 and is intended to be deployed on the Kompai platform resulting from

WP2. The SCC is integrated with the other components as part of the activities in WP7 and will be validated in the context of WP8.

1.5. Document Outline

This work takes into account the following main sections: Motivation, Semantic Correction Component, Model Description and Software, which provides a complete description of the model and of the software component.

1.6. About MARIO

MARIO addresses the difficult challenges of loneliness, isolation and dementia in older persons through innovative and multi-faceted inventions delivered by service robots. The effects of these conditions are severe and life-limiting. They burden individuals and societal support systems. Human intervention is costly but the severity can be prevented and/or mitigated by simple changes in self-perception and brain stimulation mediated by robots.

From this unique combination, clear advances are made in the use of semantic data analytics, personal interaction, and unique applications tailored to better connect older persons to their care providers, community, own social circle and also to their personal interests. Each objective is developed with a focus on loneliness, isolation and dementia. The impact centres on deep progress toward EU scientific and market leadership in service robots and a user driven solution for this major societal challenge. The competitive advantage is the ability to treat tough challenges appropriately. In addition, a clear path has been developed on how to bring MARIO solutions to the end users through market deployment.

2. Motivation

A key contribution of the MARIO project is to advance the state-of-the-art of Human-Robot-Interaction (HRI) by supporting natural language interaction between users and MARIO, allowing them to issue queries and commands over the robot Knowledge Base and set of applications.

Despite the advanced maturity of Automatic Speech Recognition (ASR) techniques, MARIO operates on environments which can be rich on background noise, containing multiple speakers and interacting with users with different levels of speech impairment (as a result of multiple levels of dementia).

Current state-of-the-art ASR tools have a limited operation under these environments conditions limiting the interpretation capabilities of the MARIO platform. The ASR step is the first step in the semantic interpretation of user intents, and an incorrect ASR step jeopardises all the subsequent semantic interpretation steps and, as a consequence, the overall quality of interaction with the MARIO platform.

This task aims at maximizing the robustness of the ASR component, correcting under a best-effort scenario, the output of the ASR component using a set of additional semantic and phonological techniques.

This deliverable describes the experimental and software development work for the elaboration of the semantic correction component.

3. Semantic Correction Component (SCC)

The semantic correction component operates under the assumption that the output of the ASR component can be improved by taking into account the lexical knowledge embedded in the MARIO Knowledge Base and in the API base.

MARIO is expected to operate under a limited (but continuously extensible) domain scenario where users have an enumerable set of queries and operations which can be issued at the platform. The term *limited* in this case means that MARIO can only target elements which are within its knowledge base. The conceptual scoping entailed by the knowledge base can be used to approximate possible misinterpretations from the ASR component, by using a combination of semantic and phonological approximation techniques.

4. Model Description

4.1. Problem Description

The semantic correction component uses the labels of the entities, query templates and *action frames* within the MARIO knowledge base as the semantic grounding element to which all natural language input needs to be matched.

Underlying the usability assumptions of MARIO lies the fact that MARIO should be robust to vocabulary and abstraction variations (user expressing similar intents with different terms). This introduces a further challenge in the correction processing, which should be able to handle an open set of lexicalizations for the MARIO KB.

4.2. Distributional-Phonological Semantic Approximation Model

Let *e* be the set of lexicalizations for the entities within the KB. These entities can correspond to *constants*, *predicates*, *actions* and *action arguments descriptions* present in the KB. For each element *e* we define two associated elements: e_p for the phonological representation of *e* and es for a distributional semantic vector representation of *e*.

Given an input term te_i within the user natural language input t = te_0, te_1, … te_k, the goal is to define an approximation function sigma which maps the set of te_i to KB entities e_k. The output of the approximation function is a set of alignments or mappings m between te_i and e_k.

The entities e_k can be predicates (classes and properties) or constants (named entities / instances, values) in the case of queries over an RDF knowledge base or action names or parameters in the case of a frame description for a command.

High-level Components

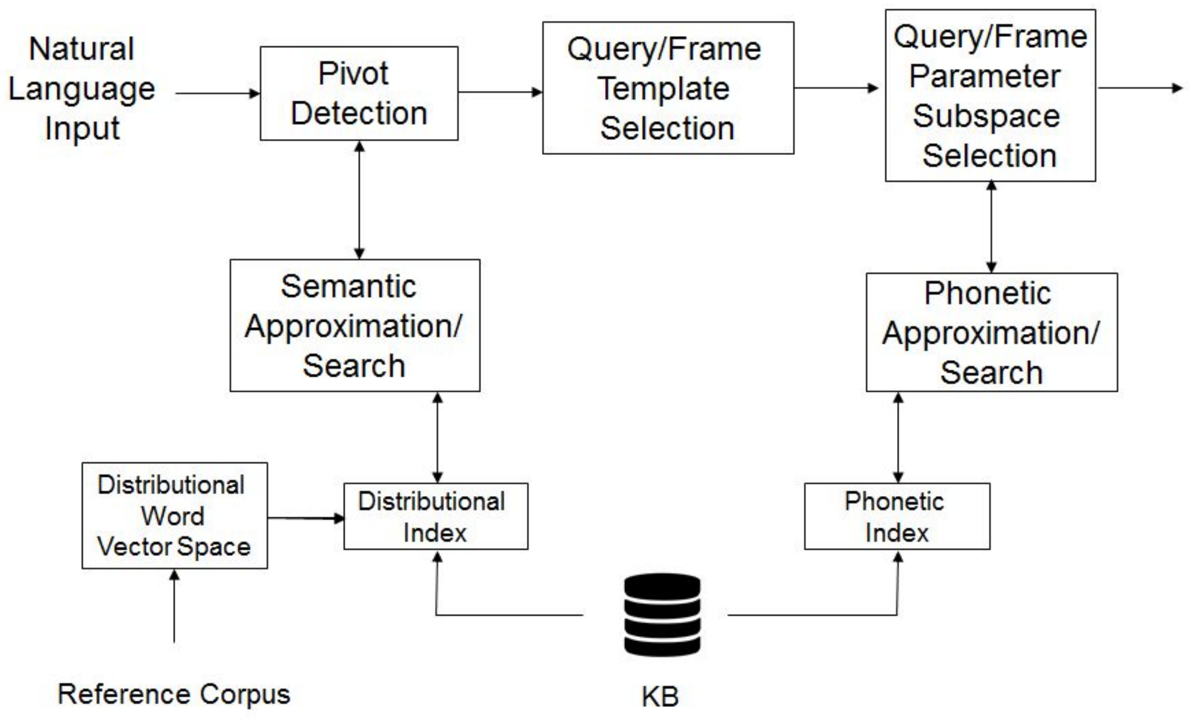
Natural Language Input
Pivot Detection
Query/Frame Template Selection
Query/Frame Parameter Subspace Selection
Semantic Approximation/ Search
Phonetic Approximation/ Search
Distributional Word Vector Space
Distributional Index
Phonetic Index
Reference Corpus
KB

Figure 1: Components of the approach of the semantic correction component.

4.3. Semantic Pivoting Model

The core rationale behind the proposed model is the use of the concept of semantic pivoting. Semantic pivoting is a generic term used to define heuristic models to address semantic mapping tasks such as the resolution of schema-agnostic queries and machine translation [2]. In general terms it consists of defining a heuristic model which can be used for reducing the semantic complexity [3] of a semantic mapping task, prioritizing mappings which are easier to address and using this first mappings as a constraint to reduce the search space for the subsequent mappings.

The definition of a heuristic model to identify the easier mappings (semantic pivots) allows the minimization of the error in the semantic matching, where a higher probability mapping is used to constrain the search space of the subsequent mappings.

Phonetic Representation

There are several algorithms which convert words into their phonetic representations. These algorithms aim at capturing possible misspellings and can be both generic and language-specific. Examples of phonetic representation algorithms are Metaphone, Double Metaphone [1].

Distributional Vector Representation

A distributional semantic model is used to cope with the vocabulary gap between the user utterances and the entity lexicalizations within the dataset. Distributional semantic models [3] supports a construction of a semantic model automatically from large-scale corpora. This allows a solution which can be transported across multiple languages (in the context of the MARIO project, English and Italian). In this work the Word2Vec/Skipgram distributional model [4] is used, using Wikipedia as a commonsense corpus.

Learning to Pivot

The semantic pivot is learned by the application of a machine learning process using features derived from the distributional vector representation of input terms te_k and KB terms e_i as well as additional linguistic features such as part of speech (POS) tags.

The pivot determination process starts by defining a set of possible/candidate mappings m_c using a broader approximation function \rho defined over the distributional space.

The distributional space provides a high recall/multi-sense mapping function, in which we can use large commonsense knowledge extracted from large-scale corpora, to define which terms from te_k can have a certain degree of distributional semantic relatedness to the e_i terms present in the KB. As the distributional model captures a multisense cross product for the lemmas present in te_k and e_i, *r* will tend to provide a high recall/low precision mapping function, assuming that the corpus constraints a representative distribution from these senses.

The set of m_c mappings are used as an input for the pivot classification step, which uses the following features for classifying every mapping/alignment:

- Density of the distributional space around e_i.
- POS tags of te_k, e_i

- Distributional semantic relatedness value between te_k and e_i.

Using an existing gold standard, containing a set of natural language inputs and the associated entities, queries or KB frames, the pivot model can be trained in order to learn high probability mappings. The rationale behind the proposed model is the combination of the density over the distributional space, the degree of semantic relatedness for a given type of alignment specified by its lexical categories can be informative indicators on the probability of an alignment.

In case of a KB query, the pivoting is performed at predication-level (at the entities in the KB) while for a command frame matching the pivoting is performed at the *action frames*. There are two main high-level heuristics for prioritizing the pivoting of action frames and predication-level entities: (i) the fact that the number of predicates/actions is typically much smaller than the number of instances/values (ii) the usually lower accuracy of ASR systems to classify named entities.

After the pivot predicate is selected, the set of queries or action frames containing that pivot element is short-listed.

Slot Filling

The next step consists in resolving the parameters of the query or action frame, i.e., resolving the constants to entities in the KB. As predicates/actions frames have associated set of elements that they are applicable to (their associated domain/range), this information is used to reduce the search space for the additional elements.

In this case, two types of approximations are possible: a set of synonymic approximations (WordNet-supported) and phonological approximations. The phonological approximation is defined by the Metaphone3 method, which converts all the elements within then KB to a phonological representation. This phonological representation allows the computation of a phonetic distance between words.

Given a certain input term te_k, we can then search the set of KB entities e_k within the predicate/action domain/range space by a set of similar sounds, using an edit distance over the Metaphone3 representation of the word.

4.4. Experiments & Evaluation

The platform was initially evaluated using as an experimental setting over the DBpedia dataset with the Question Answering over Linked Data (QALD-4) test collection. The QALD-4 test collection contains 200 natural language queries in English which are mapped to SPARQL queries over DBpedia.

The approach was evaluated by training the model using a ASR query generator (ASR-QG). The simulator takes a query from the test collection, expanding it with a set of WordNet synonyms and generating random word misinterpretations. The misinterpretations are generated after a small set of random edits are produced in a query string which is cast to a similar string from the lexicon (the combined WordNet + DBpedia lexicon) using Levenshtein distance. A set of 4000 queries were generated which were split 80% for training and 20% for test.

Different machine learning methods were compared for the semantic pivoting method, including the Multi-Layer Perceptrons (MLP), Long Short-Term Memories (LSTM), Random Forest using different combinations of the following features: lexical features, POS tags, distributional density, IDF (inverse document frequency) of the query term, mapping semantic relatedness.

From the different experiment settings, random forest using POS Tags, semantic relatedness and distributional density proved to achieved higher accuracy, achieving 85.18% accuracy on the test set.

Classif. Approaches	Basic set		
Classifier	% answered	% correct	Average nb
MLP	59.89	82.82	1.09
Random Forest	75.46	80.94	1.61
Ranker	100	65.92	1
Linear Classifier	53.83	70.59	0.7
	With Levenshtein		
Classifier	% answered	% correct	Average nb
MLP	62.66	85.05	1.13
Random Forest	75.06	82.07	1.37
Ranker	100	66.05	1
Linear Classifier	59.37	87.33	0.95
	With Embeddings		

Classifier	% answered	% correct	Average nb
MLP	46.96	81.74	0.86
Random Forest	39.71	88.7	0.7
Ranker	100	49.93	1
Linear Classifier	-	-	-
	With idf		
Classifier	% answered	% correct	Average nb
MLP	61.87	85.29	0.93
Random Forest	70.45	81.65	1.23
Ranker	100	67.77	1
Linear Classifier	52.77	88	0.75
	With mean density		
Classifier	% answered	% correct	Average nb
MLP	64.12	85.18	1.16
Random Forest	69.26	81.33	1.27
Ranker	100	67.11	1
Linear Classifier	62	83.83	0.92

Table 1: Summary of the baselines for experimental results.

5. Software

5.1. Repository

The semantic correction component can be found at */trunk/passau/semantic_corrector* in the MARIO repository.

5.2. Usage

A command line application can be invoked using `*python corrector.py --lang "en" --sentences "sentence"* . The system will return the corrected sentence.

The dependencies are described at the *"requirements.txt"* file, putting the required files into *"data/"* and *"libs/"*.

Before applying the corrector, a training model needs to be generated based on the knowledge base files and on the frame specification file. The training can be invoked using `*python training.py --lang "en" --kb "kb.nt" --queries "queries.sparql" --api "apis.frame"*, where *kb.nt* contains all the KB data, queries.sparql contains the sparql query templates and apis.frame contains the frame descriptions.

Query:

```
"sentence": "Who were the people that visited me today?",
"query": "
        PREFIX : <http://mario-project.org/onto/>
        PREFIX res: <http://mario-project.org/res/>
        SELECT ?name
        WHERE {
              res:user_co_reference :visit ?visit .
              ?visit :name ?name .
              ?visit :date "23.10.2016" .
        }
"
```

Action frame representation:

```
 {
  "data": [
   {
    "desc": null,
    "name": "image_url",
    "sample": null
   },
   {
    "desc": null,
    "name": "upload_date",
    "sample": null
   },
   {
    "desc": null,
    "name": "provided_by",
    "sample": null
   },
   {
    "desc": null,
    "name": "license",
    "sample": null
   }
  ],
  "desc": "returns the first image that match the query.",
  "id": 700006,
  "more": {
   "provider": "Flickr"
```

```
},
```

Listing 1: Representation of the SPARQL query and the command frame.

The existing code is accessible as a service and the repository contains an integrated wrapper of the python code (REST client) to the MARIO publish/subscribe middleware.

6. Dependencies

The Semantic Correction Component (SCC) was developed in Python 3.5 and depends on the following open source modules:

elasticsearch (5.0.1)
gensim (0.13.3)
joblib (0.10.3)
nltk (3.2.1)
numpy (1.11.2)
pandas (0.19.1)
scikit-learn (0.18.1)
requests (2.12.2)

Additionally, it requires a custom built version of Stanford CoreNLP and its models.

7. References

1. Lawrence Philips, Hanging on the Metaphone, Computer Language, Vol. 7, No. 12 (December), 1990.
2. André Freitas, Juliano Efson Sales, Siegfried Handschuh, Edward Curry, How hard is the Query? Measuring the Semantic Complexity of Schema-Agnostic Queries, In Proceedings of the 11th International Conference on Computational Semantics (IWCS), London, 2015.
3. André Freitas, Schema-agnostic queries over large-schema databases: a distributional semantics approach, PhD Thesis (2015).
4. Mikolov et al., Distributed Representations of Words and Phrases and their Compositionality, NIPS, (2013).