Exploiting Distributional Semantic Models in Question Answering

Piero Molino*† and Pierpaolo Basile*† and Annalina Caputo* and Pasquale Lops* and Giovanni Semeraro*

*Dept. of Computer Science - University of Bari Aldo Moro
Via E. Orabona, 4 - 70125 Bari (ITALY)
Email: piero.molino@uniba.it,{basilepp,acaputo,lops,semeraro}@di.uniba.it

†QuestionCube s.r.l.
Via Zanardelli 47, 70125 Bari (ITALY)
Email: {piero.molino,pierpaolo.basile}@questioncube.com

***Abstract*—This paper investigates the role of Distributional Semantic Models (DSMs) in Question Answering (QA), and specifically in a QA system called QuestionCube. QuestionCube is a framework for QA that combines several techniques to retrieve passages containing the exact answers for natural language questions. It exploits Information Retrieval models to seek candidate answers and Natural Language Processing algorithms for the analysis of questions and candidate answers both in English and Italian. The data source for the answer is an unstructured text document collection stored in search indices.**

In this paper we propose to exploit DSMs in the QuestionCube framework. In DSMs words are represented as mathematical points in a geometric space, also known as *semantic space*. Words are similar if they are close in that space. Our idea is that DSMs approaches can help to compute relatedness between users' questions and candidate answers by exploiting paradigmatic relations between words. Results of an experimental evaluation carried out on CLEF2010 QA dataset, prove the effectiveness of the proposed approach.

I. Introduction

Distributional Semantics Models (DSMs) represent word meanings through linguistic contexts. The meaning of a word can be inferred by the linguistic contexts in which the word occurs. The philosophical insight of distributional models can be ascribed to Wittgenstein's quote *"the meaning of a word is its use in the language"* [1]. The idea behind DSMs can be summarized as follows: if two words share the same linguistic contexts they are somehow similar in the meaning. For example, analyzing the sentences "drink wine" and "drink beer", we can assume that the words "wine" and "beer" have similar meaning. Using that assumption, the meaning of a word can be expressed by the geometrical representation in a *semantic space*. In this space a word is represented by a vector whose dimensions correspond to linguistic contexts surrounding the word. The word vector is built analyzing (e.g. counting) the contexts in which the term occurs across a corpus. Some definitions of contexts may be the set of co-occurring words in a document, in a sentence or in a window of surrounding terms.

Generally, semantic spaces do not require specific text operations, only tokenization is needed, and they are corpus/language independent. This has benefited their practical use in many different tasks. The earliest and simplest formulation of such a space has root in the Vector Space Model in Information Retrieval (IR) [2]. Since then, several linguistic and cognitive tasks have exploited semantic spaces, such as synonym choice [3], semantic priming [3]–[5], automatic construction of thesauri [6] and word sense induction [7].

This paper aims at exploiting DSMs for performing a task to which they have never been applied before, i.e. Question Answering (QA), exploring how to integrate them inside a pre-existent QA system. Our insight is based on the ability of these spaces to capture paradigmatic relations between words which should result in a list of candidate answers related to the user's question.

Question Answering (QA) emerged in the last decade as one of the most promising fields in Artificial Intelligence, as highlighted by the organization of several competitions in international conferences [8], [9], but the first studies can be dated back to 1960s [10], [11]. In recent years some enterprise applications have shown the potential of the state-of-the-art technology, such as the IBM's Watson/DeepQA system [12], [13]. By exploiting techniques borrowed from IR and Natural Language Processing (NLP), QA systems are able to answer users' questions expressed in natural language. Differently from search engines, which return long lists of full-text documents that users have to check in order to find the information needed, QA systems are able to answer users' questions with either directly the exact answer or with short passages of text containing the exact answer.

In order to test the effectiveness of the DSMs for QA, we rely on a pre-existent QA framework called QuestionCube [14]. QuestionCube is a general framework for building QA systems with focus on closed domains, but which can be easily applied to open domains as well. QuestionCube exploits NLP algorithms, for both English and Italian, in order to analyze questions and documents with the purpose of allowing candidate answers obtained from the retrieved documents to be re-ranked by a pipeline of filters. These filters assign a score to a candidate answer taking into account several linguistic and semantic features.

Our strategy for exploiting DSMs consists in adding a new filter to this pipeline, based on vector spaces built using DSMs. In particular, we propose four types of spaces: a classical Term-Term co-occurrence Matrix (TTM) used as baseline, Latent Semantic Analysis (LSA) applied to TTM, Random Indexing (RI) approach to reduce TTM dimension, and finally an approach which combines LSA and RI. The filter will assign a score based on the similarity between the question and the candidate answers inside the DSMs.

The paper is structured as follows. Section II describes how semantic spaces are built, while Section III provides a generic overview of the QuestionCube architecture. Section IV provides details about the integration of DSMs in the QuestionCube framework. Results of the evaluation are reported in Section V, while a brief overview of related work is reported in Section VI. Conclusions close the paper.

II. Distributional Semantic Models

Our DSMs are constructed over a co-occurrence matrix. The linguistic context taken into account is a window w of co-occurring terms. Given a reference corpus[1] and its vocabulary V, a $n \times n$ co-occurrence matrix is defined as the matrix $\mathbf{M} = (m_{ij})$ whose coefficients $m_{ij} \in \mathbb{R}$ are the number of co-occurrences of the words t_i and t_j within a predetermined distance w.

The $term \times term$ matrix $\mathbf{M}$, based on simple word co-occurrences, represents the simplest semantic space, called Term-Term co-occurrence Matrix (TTM).

In literature, several methods to approximate the original matrix by rank reduction have been proposed. The aim of these methods varies from discovering high-order relations between entries to improving efficiency by reducing its noise and dimensionality. We exploit three methods for building our semantic spaces: Latent Semantic Analysis (LSA), Random Indexing (RI) and LSA over RI.

All these methods produce a new matrix $\hat{\mathbf{M}}$, which is a $n \times k$ approximation of the co-occurrence matrix $\mathbf{M}$ with n row vectors corresponding to vocabulary terms, while k is the number of reduced dimensions.

A. Latent Semantic Analysis

Latent Semantic Analysis [15] is based on the Singular Value Decomposition (SVD) of the original matrix $\mathbf{M}$. $\mathbf{M}$ is decomposed in the product of three matrices $\mathbf{U\Sigma V}^\top$, where $\mathbf{U}$ and $\mathbf{V}$ are orthonormal matrices whose columns are the *right* and *left eigenvectors* of the matrices $\mathbf{M}^\top\mathbf{M}$ and $\mathbf{MM}^\top$ respectively, while $\mathbf{\Sigma}$ is the diagonal matrix of the *singular values* of $\mathbf{M}$ placed in decreasing order. SVD can be applied to any rectangular matrix, and if r is the *rank* of $\mathbf{M}$, then the matrix $\mathbf{M}^* = \mathbf{U}^*\mathbf{\Sigma}^*\mathbf{V}^{\top *}$ of rank $k \ll r$, built choosing the top k singular values, is the best rank k approximation of $\mathbf{M}$. SVD helps both to discover high-order relations between terms and to reduce the sparsity of the original matrix. Moreover, since the matrix $\mathbf{MM}^\top$ corresponds to all possible combinations of any two terms, it is possible to compute the similarity between two terms by exploiting the relation

$$\mathbf{MM}^\top = \mathbf{U\Sigma V}^\top\mathbf{V\Sigma}^\top\mathbf{U}^\top = \mathbf{U\Sigma\Sigma}^\top\mathbf{U}^\top = (\mathbf{U\Sigma})(\mathbf{U\Sigma})^\top$$

In the case of the k-approximation of $\mathbf{M}$, the complexity of the computation of the similarity between any two terms is reduced.

B. Random Indexing

We exploit Random Indexing (RI), introduced by Kanerva [16], for creating the DSM based on RI. This technique allows us to build a semantic space with no need for matrix factorization, because vectors are inferred using an incremental strategy. Moreover, it allows to solve efficiently the problem of reducing dimensions, which is one of the key features used to uncover the “latent semantic dimensions” of a word distribution.

RI is based on the concept of Random Projection according to which randomly chosen high dimensional vectors are “nearly orthogonal”.

Formally, given an $n \times m$ matrix $\mathbf{M}$ and an $m \times k$ matrix $\mathbf{R}$ made up of m k-dimensional random vectors, we define a new $n \times k$ matrix $\mathbf{M}'$ as follows:

$$\mathbf{M'}_{n,k} = \mathbf{M}_{n,m}\mathbf{R}_{m,k} \quad k << m \tag{1}$$

The new matrix $\mathbf{M}'$ has the property to preserve the distance between points. This property is known as Johnson-Lindenstrauss lemma [17]: if the distance between any two points of $\mathbf{M}$ is d, then the distance d_r between the corresponding points in $\mathbf{M}'$ will satisfy the property that $d_r = c \cdot d$. A proof of that property is reported in [18].

The product between $\mathbf{M}$ and $\mathbf{R}$ is not actually computed, but it corresponds to building $\mathbf{M}'$ incrementally, as follows:

1) Given a corpus, a *random vector* is assigned to each term. The random vector is high-dimensional, sparse and with very few elements with non-zero values $\{-1, 1\}$, which ensures that the resulting vectors are nearly orthogonal, and the structure of this vector follows the hypothesis behind the concept of Random Projection.
2) The *semantic vector* of a term is given by summing the random vectors of terms co-occurring with the target term in a predetermined context (document/sentence/window).

C. Latent Semantic Analysis over Random Indexing

Computing LSA on the co-occurrence matrix $\mathbf{M}$ can be a computationally expensive task, as the vocabulary V can reach thousands of terms. Here we propose a simpler computation based on the application of the SVD factorization to $\mathbf{M}'$, the reduced approximation of $\mathbf{M}$ produced by Random Indexing. Sellberg and Jönsson [19] followed a similar approach for the retrieval of similar FAQs in a QA system. Their experiments showed that reducing the original matrix by RI resulted in a drastic reduction of LSA computation time. The trade-off to be paid was the slight worse performance, which were better than TTM and RI anyway.

[1]In our case the collection of documents indexed by the QA system.

III. Framework overview

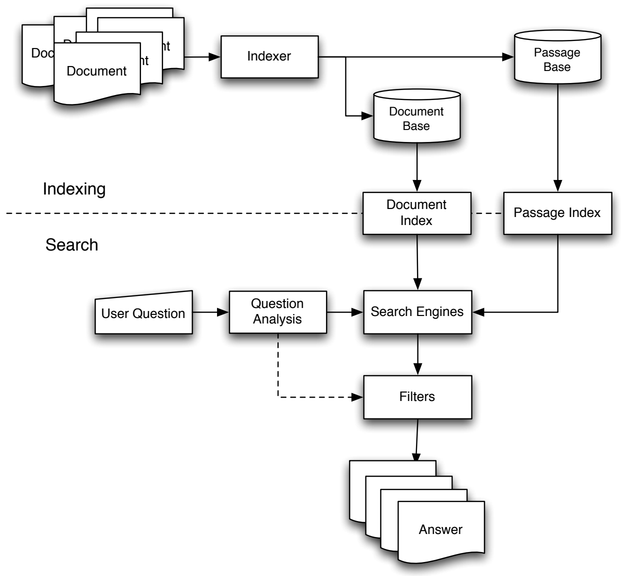

Fig. 1. QuestionCube architecture overview

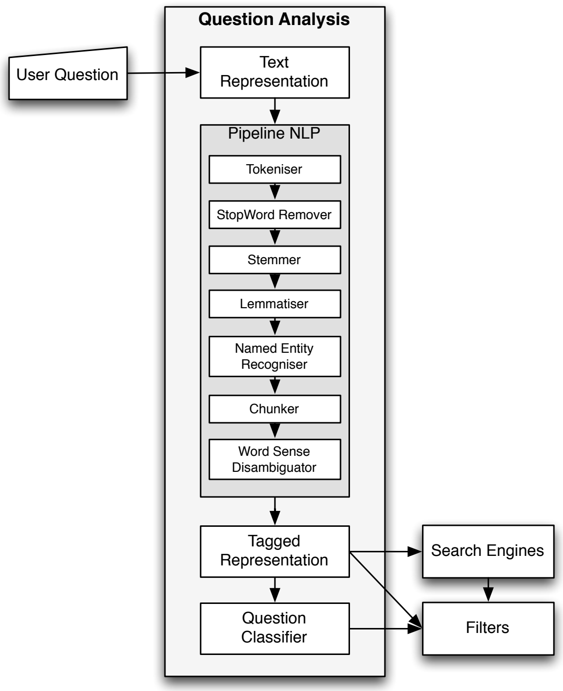

Fig. 2. Question analysis macro-component

QuestionCube is a multilingual QA framework built using NLP and IR techniques. The main aim of the QuestionCube architecture, shown in Figure 1, is to create a QA system simply by the dynamic composition of framework components, as suggested in [20].

Question Answering is a two steps process. At indexing time, the system builds two different indexes for documents and for passages contained in each document. At query time, the user's question is analyzed by a NLP pipeline. The result of this pipeline is a text tagged with linguistic annotation useful for passage re-ranking. The question is then passed to the search engines. Then, the filter pipeline is responsible for the filtering and scoring of the passages retrieved by search engines. Finally, the ranked list of passages is presented to the user.

A. Question Analysis

The macro-component of the question analysis consists of a pipeline of NLP analyzers, a data-structure to represent linguistic annotated text and the question classifier, as shown in Figure 2. The NLP pipeline is easily configurable depending on the application domain of the QA system.

NLP analyzers are provided for both English and Italian. The stemmer is implemented by Snowball[2] both for English and Italian. The lemmatiser is realized by exploiting the morpho-syntactic analyzer of WordNet API [21] for English, while Morph-it [22] is exploited for Italian. Named Entity Recognition (NER) is performed by a machine learning classifier based on Support Vector Machines [23] using an open-source tool called YAMCHA [24]. The same tool is used for the chunker component. POS-tags and lemmas are adopted as features for chunking and NER. The Word Sense Disambiguation (WSD) is implemented by the UKB algorithm [25], which is a graph-based technique based on a personalized version of PageRank [26] over WordNet graph.

The output of the NLP analyzers is a set of tags that are added to the text representation.

The annotated text representation of the question is used by the question classifier. It exploits both a machine learning approach (support Vector Machines) and hand-written rules. For more details refer to [14].

The category is selected among those from the typology proposed in [27], [28]. Categories are exploited by filters in order to give higher scores to those candidate answers containing Named Entities, in accordance with the question category.

B. Search Engine

The search engine macro-component is designed to allow the integration of several information retrieval strategies, and thus the aggregation of their results, as shown in Figure 3. Since the parallel searcher enables modularity, it is possible to add an arbitrary number of different search engines. When a new question comes, the parallel searcher calls each engine and merges their outputs in a single list. The list contains all the candidate answers collected from all the engines, each one with a reference to the engines that retrieved it and the score assigned by each engine. Each search engine has its own query generation component, as the query syntax can change among different engines. Moreover, each query generator can

[2]Available on-line: http://snowball.tartarus.org/

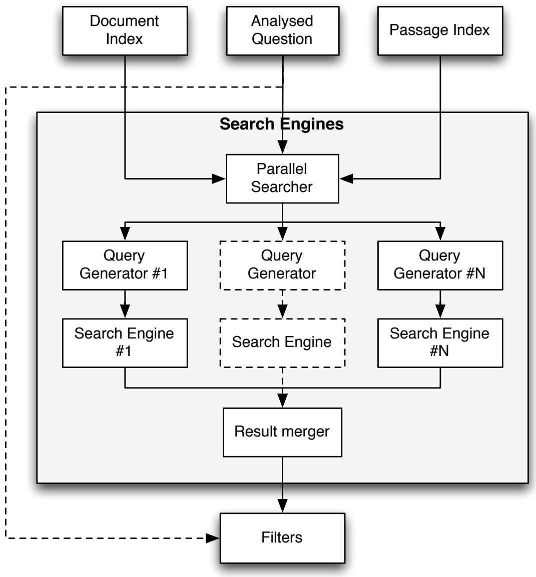

Fig. 3. Search engine macro-component

use different annotations: some use only tokens, lemmas or stems, while other ones may use WordNet synsets to generate the query representation. This approach makes the framework highly modular and flexible and allows to add a new search engine inside the framework with minimal effort. The main goal of using more than one search engine is to rely on different retrieval strategies in order to exploit different text representations and retrieval models.

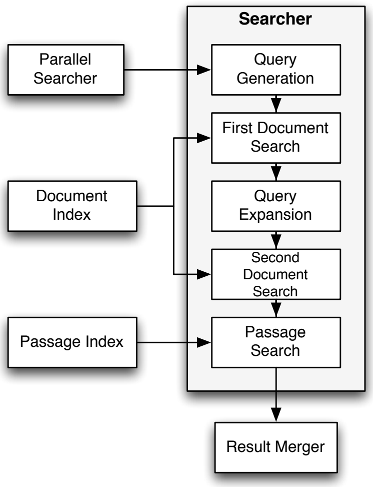

Fig. 4. Single search engine

The process performed by each search engine is described in Figure 4. Each query generator builds the query for its search engine from the text representation provided by the parallel engine. Moreover, the query generator may implement different query-improvement techniques (such as relevance feedback and query expansion). Hence, the search engine executes the query to return the best scored documents. The passage index is used to obtain passages from retrieved documents. These passages are merged into one single list by an aggregation component and then passed to the filters which score, sort and filter them.

QuestionCube provides a search engine based on *BM25* model [29] and another based on Apache Lucene[3]. The query generation component for those searchers allows three different query-improvement techniques:

- *Query expansion* through WordNet synonyms of synsets found in the question;
- *Kullback-Leibler Divergence*, a statistical technique that exploits the distribution of terms in the top-ranked documents [30], [31];
- *Divergence From Randomness*, a statistical technique that weights the terms' distribution with the Bose-Einstein *Bo1* weighting scheme [32].

It is important to underline that the WordNet-based query expansion is used only if the question has been disambiguated.

C. Filters

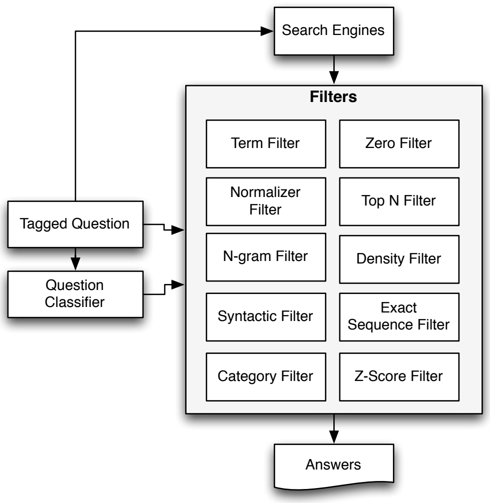

Fig. 5. Candidate answers filtering macro-component

This macro-component, sketched in Figure 5, contains all passage filters. Such pipeline is modular, and new filters can be added at any time.

A filter checks every input passage obtained from the search engine, and assigns a score depending on the implemented logic. Filters can exploit both information provided by the text representation and the question category tag assigned by the classifier.

Some filters do not assign scores, but only sort the passages according to some score or rank threshold.

The filter composition in the pipeline is important to determine the quality of results returned by the system, the

[3]Available at http://lucene.apache.org/

efficiency and the answering time. A description of each filter logic is listed below:

- **Zero Filter**: removes from the list all those passages that, at the moment of the analysis, have a score of 0;
- **Top-N Filter**: sorts and filters passages to the top N-ranked ones on the basis of their current score;
- **Terms filter**: assigns a score to a passage based on the frequency of occurring question terms;
- **Exact Sequence Filter**: assigns a score to a passage based on the number of the longest overlapping sequence of question terms found in the passage;
- **Normalization Filter**: assigns a score based on the passage length normalized by its overall score. Both a simple normalization filter (*Byte-size Normalization*, which considers only the number of terms) and a filter based on the *Pivoted Normalized Document Length* techniques are implemented. More details about these techniques are presented in [33], [34];
- **N-grams Filter**: assigns a score to a passage based on the overlapping of n-grams between the question and the passage (n is given as input to the filter);
- **Density Filter**: assigns a score to a passage based on the distance of the question terms inside it. The closer the question terms appear in the passage, the higher the score. The density is calculated by a modified version of the Minimal Span Weighting schema proposed by [35]:

  $$\left(\frac{\mid q \cap d \mid}{1 + \max(mms) - \min(mms)}\right)$$

  where q and d are the set of terms in the query and in the document, respectively (specifically, the query is the question and the document is the passage); $\max(mms)$ and $\min(mms)$ are the initial and final location of the sequence of document terms containing all the query terms.
- **Syntactic Filter**: assigns a score to a passage based on the Phrase Matching algorithm presented in [36]. The algorithm takes into account the head of each phrase. If the head is common to the two texts considered (in this case the question and the passage), the maximal overlapping length of each phrase is calculated.
- **Category Filter**: assigns a score to a passage based on a list of pairs that link the question categories to typologies of named entity: if, on the basis of the question category, entities of the expected typology are found in the passage this will get a positive score.
- **Z-Score Filter**: assigns a score to a passage based on the Z-Score normalization [37] of scores assigned by the search engines and the other filters.
- **CombSum Filter**: assigns a score to a passage summing the scores assigned by the search engines and the other filters [37] .

Terms and Density filters have an enhanced version which adopts the combination of lemmas and PoS tags as features instead of terms.

A boost factor can be assigned to each filter which intensifies or decreases its strength.

IV. DSMs integration into the QuestionCube framework

The idea behind the application of DSMs to the QuestionCube framework is to build a new filter relying on our semantic spaces, and add it to the filter pipeline as shown in Figure 5. We call this component **Distributional Filter**, which aims at computing the similarity between the user's question and each candidate answer.

In DSMs, given the vector representation of two words $\mathbf{u} = (u_1, u_2, ..., u_n)^\top$ and $\mathbf{v} = (v_1, v_2, ..., v_n)^\top$, it is always possible to compute their similarity as the cosine of the angle between them:

$$\cos(\mathbf{u}, \mathbf{v}) = \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2 \sum_{i=1}^{n} v_i^2}} \tag{2}$$

However, the user's question and the candidate answer are sentences composed by several terms, so in order to compute the similarity between them we need a method to compose the words occurring in these sentences. It is possible to combine words through vector addition $(+)$. This operator is similar to the superposition defined in connectionist systems [38], and corresponds to the point-wise sum of components:

$$\mathbf{p} = \mathbf{u} + \mathbf{v} \tag{3}$$

where $p_i = u_i + v_i$

Addition is a commutative operator, which means that it does not take into account any order or underlying structures existing between words. In this first study, we do not exploit more complex methods to combine word vectors.

Given a phrase or sentence *p*, we denote with $\mathbf{p}$ its vector representation obtained applying addition operator $(+)$ to the vector representation of terms it is composed of. Furthermore, it is possible to compute the similarity between two phrases/sentences exploiting the cosine similarity between vectors (2).

Formally, if $q = q_1 q_2 ... q_n$ and $a = a_1 a_2 ... a_m$ are the question and the candidate answer respectively, we build two vectors $\mathbf{q}$ and $\mathbf{a}$ which represent respectively the question and the candidate answer in a semantic space. Vector representations for question and answer are built applying the addition operator to the vector representation of words belonging to them:

$$\begin{aligned} \mathbf{q} &= q_1 + q_2 + \ldots + q_n \\ \mathbf{a} &= a_1 + a_2 \ldots + a_m \end{aligned} \tag{4}$$

The similarity between $\mathbf{q}$ and $\mathbf{a}$ is computed as the cosine similarity between them. This similarity is added as score to the candidate answer

V. Evaluation

The goal of the evaluation is twofold: (1) proving the effectiveness of DSMs into our question answering system and (2) providing a comparison between the several DSMs adopted by the **Distributional Filter**.

The evaluation has been performed on the *ResPubliQA 2010 Dataset* adopted in the *2010 CLEF QA Competition* [9]. The dataset contains about 10,700 documents of the European Union legislation and European Parliament transcriptions, aligned in several languages including English and Italian, with 200 questions.

The adopted metric is the accuracy $a@n$, calculated considering only the first n answers. If the correct answer occurs in the top n retrieved answers, the question is marked as correctly answered. In particular, we take into account several values of $n =$1, 5, 10 and 30. Moreover, we adopt another metric, **MRR**, that considers the rank of the correct answer. MRR is defined as follows:

$$MRR = \frac{\sum_{i=1}^{N} \frac{1}{rank_i}}{N} \quad (5)$$

where N is the number of questions and $rank_i$ is the rank of the correct answer for the $i-th$ question. The correct answer receives a higher score if it occurs on the top of the global rank.

The framework setup used for the evaluation adopts Lucene as document searcher, and stemmer, lemmatizer, PoS tagger and named entity recognition as NLP pipeline. The different DSMs and the classic TTM have been used as filters alone, which means no other filters are adopted in the filters pipeline, and combined with the standard filter pipeline consisting of the Simple Terms (ST), the Enhanced Terms (ET), the Enhanced Density (ED) and the Exact Sequence (E) filters. The combination consist in normalizing the scores with the Z-Score filter and then summing their results with the CombSum [37] filter. The performance of the standard pipeline, without the distributional filter, is shown as a baseline. The experiments have been carried out both for English and Italian, results are shown respectively in Table I and II. Each Table reports the accuracy $a@n$ computed considering a different number of answers, the MRR and the significance of the results with respect to both the baseline (†) and the distributional model based on TTM (‡). The significance is computed using the non-parametric Randomization test, as suggested in [39], since it has proven to be an effective test under several circumstances. For the Randomization test a Perl script supplied by the authors[4] has been employed.

Moreover, we need to setup some parameters of DSMs. The window w of terms considered for computing the co-occurrence matrix is 4, while the number of reduced dimensions considered in LSA, RI and LSARI is equal to 1,000.

Considering each distributional filter on its own, the results prove that all the proposed DSMs are better than the TTM, and the improvement is always significant. The best improvement in English is obtained by LSA (+180%), while in Italian by LSARI (+161%).

[4]http://www.mansci.uwaterloo.ca/∼msmucker/software/paired-randomization-test-v2.pl

TABLE I
EVALUATION RESULTS FOR ENGLISH. BASIC SEARCHER: LUCENE WITH KEYWORD. STANDARD FILTERS FOR COMBINED EVALUATION: ST+ET+ED+E. SIGNIFICANCE OF THE DIFFERENCE FOR THE 0.05 WITH THE BASELINE (†) AND WITH TTM (‡) ARE SHOWN.

	Run	a@1	a@5	a@10	a@30	MRR
alone	TTM	0.060	0.145	0.215	0.345	0.107
	RI	0.180	0.370	0.425	0.535	0.267‡
	LSA	0.205	0.415	0.490	0.600	0.300‡
	LSARI	0.190	0.405	0.490	0.620	0.295‡
combined	*baseline*	*0.445*	*0.635*	*0.690*	*0.780*	*0.549*
	TTM	0.535	0.715	0.775	0.810	0.614†
	RI	0.550	0.730	0.785	0.870	0.637†‡
	LSA	0.560	0.725	0.790	0.855	0.637†
	LSARI	0.555	0.730	0.790	0.870	0.634†

TABLE II
EVALUATION RESULTS FOR ITALIAN. BASIC SEARCHER: LUCENE WITH KEYWORD. STANDARD FILTERS FOR COMBINED EVALUATION: ST+ET+ED+E. SIGNIFICANCE OF THE DIFFERENCE FOR THE 0.05 WITH THE BASELINE (†) AND WITH TTM (‡) ARE SHOWN.

	Run	a@1	a@5	a@10	a@30	MRR
alone	TTM	0.060	0.140	0.175	0.280	0.097
	RI	0.175	0.305	0.385	0.465	0.241‡
	LSA	0.155	0.315	0.390	0.480	0.229‡
	LSARI	0.180	0.335	0.400	0.500	0.254‡
combined	*baseline*	*0.365*	*0.530*	*0.630*	*0.715*	*0.441*
	TTM	0.405	0.565	0.645	0.740	0.539†
	RI	0.465	0.645	0.720	0.785	0.555†
	LSA	0.470	0.645	0.690	0.785	0.551†
	LSARI	0.480	0.635	0.690	0.785	0.557†‡

Taking into account the distributional filters combined with the standard filter pipeline, the results prove that all the combinations are able to overcome the baseline. In English we obtain an improvement, about 16% with respect to the baseline, and the result obtained by the TTM is significant. For the Italian language, we achieve a better improvement. The best result is obtained by LSARI with an improvement of 26% with respect to the baseline.

The slight difference in performance between LSA and LSARI proves that LSA applied to the matrix obtained by RI produce the same result of the LSA applied to TTM, but requiring less computation time, as the matrix obtained by RI contains less dimensions than the TTM matrix.

In general, the obtained results are encouraging and prove the effectiveness of using DSMs in our QA framework. Moreover, the proposed *semantic spaces* are able to outperform the classical TTM in our evaluation.

Finally, the improvement obtained considering each distributional filter on its own shows an higher improvement than their combination with the standard filter pipeline. This suggests that a more complex method to combine filters should be used in order to strengthen the contribution of each of them. To this purpose, we plan to investigate some learning to rank [40] approaches from IR as future work.

VI. Related work

Most closed-domain QA systems use a variety of NLP methods to help the understanding of users' questions and the matching of passages extracted from documents [41], [42]. The most commonly adopted linguistic analysis steps include: stemming, lemmatization with dictionaries, part-of-speech tagging, parsing, named entity recognition, lexical semantics (Word Sense Disambiguation), etc. The use of these NLP steps is fundamental to find the correct answer in closed-domain QA, since there is likely to be few answers to any user's question and the way in which they are expressed may be significantly different from the question, as stressed in [43]. The difficulty of the task lies in mapping questions to answers by way of uncovering complex lexical, syntactic, or semantic relationships between questions and candidate answers.

On the other hand, Open-domain QA systems exploit redundancy alongside with textual pattern extraction and matching to find and rank candidate answers [44]–[47].

Semantic approaches have been adopted among QA systems to improve performance. The Ephyra framework [48] leveraged Semantic Role Labeling to identify semantic structures in documents that match those in the question. Semantic information has shown to be useful also in the re-ranking of passages as shown in [49]. However, semantic approaches can be exploited to provide an enriched visual representation of the answer in the form of a semantic graph, as described by [50].

Our approach differs from the proposed literature as it exploits both explicit semantic information obtained from the NLP pipeline and latent semantic information coming from DSMs. Moreover, our system supports different languages and implements all the analysis and searching steps for both English and Italian. Indeed, it is important to underline that the proposed DSMs are language-independent.

DSMs have not been used directly in question answering, while some applications to IR exist. In [51] the authors describe a tool which relies on Random Indexing (RI) to built an IR model. LSA is widely used in IR to perform term-doc matrix reduction obtaining good results and significant improvement with respect to the classical Vector Space Model [15]. In [6] an approach to ambiguity resolution in IR is proposed. The authors describe a sense-based retrieval, a modification of the standard vector-space model, in which the meanings of a word are inferred by applying clustering technique to a word space. The similarity between the word vector and the closest cluster centroid will give the proper sense. The IR system proposed by the authors gives an improvement in precision with respect to the word-based retrieval. Moreover, in [52], an IR system able to combine word sense disambiguation (WSD) and word sense discrimination based on RI is proposed. The combination of WSD and RI is performed by a semantic engine, SENSE, which is able to combine several document representations in a unique framework.

DSMs are widely exploited in the computational linguistic field in order to solve problems related to word similarity and semantic composition of meanings. A useful survey of the use of VSMs for semantic processing of text is reported in [53], while an analysis of some compositional operators is described in [54].

VII. Conclusions

In this paper, a method to integrate Distributional Semantic Models (DSMs) into a Question Answering system, called QuestionCube, has been described. DSMs represent words as mathematical points in a geometric space, also known as *semantic space*. Words are similar if they are close in that space. The idea is to exploit this kind of similarity to compute the relevance of candidate answers with respect to the user's question. Moreover, we propose several kinds of DSMs based on classical Term-Term co-occurrence Matrix (TTM), latent semantic analysis (LSA), random indexing (RI) and a combination of the last two techniques. The evaluation has been performed on the *ResPubliQA 2010 Dataset* adopted in the *2010 CLEF QA Competition* [9] both for English and Italian. The results prove the effectiveness of the proposed approach, and highlight that more sophisticated techniques such as LSA and RI are able to outperform the simple term-term matrix used as baseline.

As future work, we plan to investigate more complex operators for the vector representation of both answers and questions by exploiting the relations between terms.

References

[1] L. Wittgenstein, *Philosophical Investigations*. Blackwell, 1953, (Translated by G.E.M. Anscombe).

[2] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Commun. ACM*, vol. 18, pp. 613–620, November 1975.

[3] T. K. Landauer and S. T. Dumais, "A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge," *Psychological Review*, vol. 104, pp. 211–240, 1997.

[4] C. Burgess, K. Livesay, and K. Lund, "Explorations in context space: Words, sentences, discourse," *Discourse Processes*, vol. 25, no. 2-3, pp. 211–257, 1998.

[5] M. N. Jones and D. J. K. Mewhort, "Representing word meaning and order information in a composite holographic lexicon," *Psychological Review*, vol. 114, no. 1, pp. 1–37, 2007.

[6] H. Schütze and J. O. Pedersen, "Information retrieval based on word senses," in *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval*, 1995, pp. 161–175.

[7] H. Schütze, "Automatic word sense discrimination," *Comput. Linguist.*, vol. 24, pp. 97–123, March 1998. [Online]. Available: http://portal.acm.org/citation.cfm?id=972719.972724

[8] E. M. Voorhees and D. M. Tice, "The trec-8 question answering track evaluation," in *In Text Retrieval Conference TREC-8*, 1999, pp. 83–105.

[9] A. Penas, P. Forner, A. Rodrigo, R. F. E. Sutcliffe, C. Forascu, and C. Mota, "Overview of ResPubliQA 2010: Question Answering Evaluation over European Legislation." in *Working notes of ResPubliQA 2010 Lab at CLEF 2010*, M. Braschler, D. Harman, and E. Pianta, Eds., 2010.

[10] J. Bert F. Green, A. K. Wolf, C. Chomsky, and K. Laughery, "Baseball, an automatic question-answerer," *Managing Requirements Knowledge, International Workshop on*, vol. 0, p. 219, 1961.

[11] R. Simmons, "Answering english questions by computer: A survey," *Communications of the ACM*, vol. 8, no. 1, pp. 53–70, 1965.

[12] D. A. Ferrucci, E. W. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. M. Prager, N. Schlaefer, and C. A. Welty, "Building Watson: An Overview of the DeepQA Project," *AI Magazine*, vol. 31, no. 3, pp. 59–79, 2010.

[13] D. A. Ferrucci, "Ibm's watson/deepqa," *SIGARCH Computer Architecture News*, vol. 39, no. 3, 2011.

[14] P. Molino and P. Basile, "QuestionCube: a Framework for Question Answering," in *IIR*, ser. CEUR Workshop Proceedings, G. Amati, C. Carpineto, and G. Semeraro, Eds., vol. 835. CEUR-WS.org, 2012, pp. 167–178.

[15] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[16] P. Kanerva, *Sparse Distributed Memory*. MIT Press, 1988.

[17] W. B. Johnson and J. Lindenstrauss, "Extensions of lipschitz mappings into a hilbert space," *Conference on Modern Analysis and Probability, Contemporary Mathematics*, vol. 26, p. 189–206, 1984.

[18] S. Dasgupta and A. Gupta, "An elementary proof of the Johnson-Lindenstrauss lemma," Technical Report TR-99-006, International Computer Science Institute, Berkeley, California, USA, Tech. Rep., 1999.

[19] L. Sellberg and A. Jönsson, "Using random indexing to improve singular value decomposition for latent semantic analysis," in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC2008)*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, and D. Tapias, Eds. Marrakech, Morocco: European Language Resources Association (ELRA), 2008, pp. 2335–2338.

[20] N. Schlaefer, P. Gieselman, and G. Sautter, "The ephyra qa system at trec 2006," in *TREC*, 2006.

[21] C. Fellbaum, *WordNet: an electronic lexical database*, ser. Language, speech, and communication. MIT Press.

[22] E. Zanchetta and M. Baroni, "Morph-it! a free corpus-based morphological resource for the italian language," *Corpus Linguistics 2005*, vol. 1, no. 1, 2005.

[23] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[24] T. Kudo and Y. Matsumoto, "Fast Methods for Kernel-Based Text Analysis," in *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*. Sapporo, Japan: ACL, July 2003, pp. 24–31.

[25] E. Agirre and A. Soroa, "Personalizing PageRank for word sense disambiguation," in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, ser. EACL '09. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 33–41.

[26] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," in *Seventh International World-Wide Web Conference (WWW 1998)*, 1998.

[27] X. Li and D. Roth, "Learning question classifiers," in *Proceedings of the 19th international conference on Computational linguistics - Volume 1*, ser. COLING '02. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp. 1–7.

[28] ——, "Learning question classifiers: the role of semantic information," *Nat. Lang. Eng.*, vol. 12, pp. 229–249, September 2006.

[29] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Found. Trends Inf. Retr.*, vol. 3, pp. 333–389, April 2009.

[30] C. Carpineto, R. de Mori, G. Romano, and B. Bigi, "An information-theoretic approach to automatic query expansion," *ACM Trans. Inf. Syst.*, vol. 19, pp. 1–27, 2001.

[31] Y. Lv and C. Zhai, "Positional relevance model for pseudo-relevance feedback," in *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '10. New York, NY, USA: ACM, 2010, pp. 579–586.

[32] G. Amati and C. J. Van Rijsbergen, "Probabilistic models of information retrieval based on measuring the divergence from randomness," *ACM Trans. Inf. Syst.*, vol. 20, pp. 357–389, October 2002.

[33] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[34] A. Singhal, C. Buckley, and M. Mitra, "Pivoted document length normalization," in *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '96. New York, NY, USA: ACM, 1996, pp. 21–29.

[35] C. Monz, "Minimal span weighting retrieval for question answering," in *Proceedings of the SIGIR Workshop on Information Retrieval for Question Answering*, R. Gaizauskas, M. Greenwood, and M. Hepple, Eds., 2004, pp. 23–30.

[36] C. Monz and M. de Rijke, "Tequesta: The university of amsterdam's textual question answering system," in *TREC*, 2001.

[37] J. A. Shaw, E. A. Fox, J. A. Shaw, and E. A. Fox, "Combination of multiple searches," in *The Second Text REtrieval Conference (TREC-2*, 1994, pp. 243–252.

[38] P. Smolensky, "Tensor product variable binding and the representation of symbolic structures in connectionist systems," *Artificial Intelligence*, vol. 46, no. 1-2, pp. 159–216, Nov. 1990. [Online]. Available: http://dx.doi.org/10.1016/0004-3702(90)90007-M

[39] M. D. Smucker, J. Allan, and B. Carterette, "A comparison of statistical significance tests for information retrieval evaluation," in *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. New York, NY, USA: ACM, 2007, pp. 623–632.

[40] T. Liu, "Learning to rank for information retrieval," *Foundations and Trends in Information Retrieval*, vol. 3, no. 3, pp. 225–331, 2009.

[41] S. M. Harabagiu, D. I. Moldovan, M. Pasca, R. Mihalcea, M. Surdeanu, R. C. Bunescu, R. Girju, V. Rus, and P. Morarescu, "Falcon: Boosting knowledge for answer engines," in *TREC*, 2000.

[42] E. H. Hovy, L. Gerber, U. Hermjakob, M. Junk, and C.-Y. Lin, "Question answering in webclopedia," in *TREC*, 2000.

[43] J. Chen, A. Diekema, M. D. Taffet, N. J. McCracken, N. E. Ozgencil, O. Yilmazel, and E. D. Liddy, "Question answering: Cnlp at the trec-10 question answering track," in *TREC*, 2001.

[44] S. Dumais, M. Banko, E. Brill, J. Lin, and A. Ng, "Web question answering: is more always better?" in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '02. New York, NY, USA: ACM, 2002, pp. 291–298.

[45] S. M. Harabagiu, M. A. Paşca, and S. J. Maiorano, "Experiments with open-domain textual question answering," in *Proceedings of the 18th conference on Computational linguistics - Volume 1*, ser. COLING '00. Stroudsburg, PA, USA: Association for Computational Linguistics, 2000, pp. 292–298.

[46] M. Paşca, *Open-domain question answering from large text collections*, ser. Studies in computational linguistics. CSLI Publications, 2003.

[47] J. Lin, "An exploration of the principles underlying redundancy-based factoid question answering," *ACM Trans. Inf. Syst.*, vol. 25, April 2007.

[48] N. Schlaefer, J. Ko, J. Betteridge, M. A. Pathak, E. Nyberg, and G. Sautter, "Semantic Extensions of the Ephyra QA System for TREC 2007," in *TREC*, 2007.

[49] M. W. Bilotti, "Linguistic and semantic passage retrieval strategies for question answering," Ph.D. dissertation, Carnegie Mellon University, 2009.

[50] L. Dali, D. Rusu, B. Fortuna, D. Mladenic, and M. Grobelnik, "Question answering based on semantic graphs," in *Proceedings of the Workshop on Semantic Search (Sem-Search 2009)*, 2009.

[51] D. Widdows and K. Ferraro, "Semantic Vectors: A Scalable Open Source Package and Online Technology Management Application," in *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC2008)*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, and D. Tapias, Eds. Marrakech, Morocco: European Language Resources Association (ELRA), 2008, pp. 1183–1190.

[52] P. Basile, A. Caputo, and G. Semeraro, "Integrating Sense Discrimination in a Semantic Information Retrieval System," in *Information Retrieval and Mining in Distributed Environments*, ser. Studies in Computational Intelligence, A. Soro, E. Vargiu, G. Armano, and G. Paddeu, Eds. Springer Berlin / Heidelberg, 2011, vol. 324, pp. 249–265.

[53] P. D. Turney and P. Pantel, "From frequency to meaning: vector space models of semantics," *J. Artif. Int. Res.*, vol. 37, no. 1, pp. 141–188, Jan. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1861751.1861756

[54] J. Mitchell and M. Lapata, "Composition in distributional models of semantics," *Cognitive Science*, vol. 34, no. 8, pp. 1388–1429, 2010. [Online]. Available: http://dx.doi.org/10.1111/j.1551-6709.2010.01106.x