Towards an Approximative Ontology-Agnostic Approach for Logic Programs

João Carlos Pereira da Silva[1] and André Freitas[2]

[1] Departamento de Ciência da Computação - Instituto de Matemática - Universidade Federal do Rio de Janeiro - Brazil
[2] DERI National University of Ireland, Galway
jcps@dcc.ufrj.br,
andre.freitas@deri.org

Abstract. Distributional semantics focuses on the automatic construction of a semantic model based on the statistical distribution of co-located words in large-scale texts. Deductive reasoning is a fundamental component for semantic understanding. Despite the generality and expressivity of logical models, from an applied perspective, deductive reasoners are dependent on highly consistent conceptual models, which limits the application of reasoners to highly heterogeneous and open domain knowledge sources. Additionally, logical reasoners may present scalability issues. This work focuses on advancing the conceptual and formal work on the interaction between distributional semantics and logic, focusing on the introduction of a distributional deductive inference model for large-scale and heterogeneous knowledge bases. The proposed reasoning model targets the following features: (i) an approximative ontology-agnostic reasoning approach for logical knowledge bases, (ii) the inclusion of large volumes of distributional semantics commonsense knowledge into the inference process and (iii) the provision of a principled geometric representation of the inference process.

Keywords: distributional semantics, logic programming, knowledge bases, distributional vector space, approximate deductive reasoning.

1 Introduction

Logical models provide a comprehensive system for representing concepts, objects, their properties and associations. In addition to the representation of conceptual abstractions, logical models provide a precise definition of logical inference, allowing new knowledge to become explicit from existing facts and rules.

Despite the fundamental importance of inference to the development of intelligent systems, experimental research over large-scale and heterogeneous knowledge bases shows evidence that logical models have limitations in the provision of inference models which can cope with the level of contextual complexity, vagueness, ambiguity and scale present in open domain/commonsense knowledge bases. The lack of properties such as robustness to inconsistencies, a more principled mechanism of semantic approximation and the ability to scale to large

C. Beierle and C. Meghini (Eds.): FoIKS 2014, LNCS 8367, pp. 415–432, 2014.

volume knowledge bases represents a solid barrier to the applicability of existing inference models into this scenario.

More recently, *distributional semantic models* (DSMs) [17] have emerged from the empirically supported evidence that semantic models automatically derived from statistical co-occurrence patterns on large corpora provide simplified but comprehensive semantic models. With the availability of large volumes of text on the Web, DSMs have the potential to become a fundamental element in addressing existing challenges for enabling a robust semantic interpretation by computers.

This work investigates the complementary aspects between distributional semantics and logic programming models, focusing on the analysis of approximate inference and querying on a *distributional vector space*. While logical models provide an expressive conceptual representation structure with support for inferences and expressive query capabilities, distributional semantics provides a complementary layer where the semantic approximation supported by large-scale comprehensive semantic models and the scalability provided by the vector space model (VSM) can address the trade-off between expressivity and semantic/terminological flexibility.

The contributions of this work concentrate on advancing the conceptual and formal work on the interaction between distributional semantics and logic, focusing on the investigation of a distributional deductive inference model for large-scale and heterogeneous knowledge bases. The proposed inference model targets the following features: (i) an approximative ontology-agnostic reasoning approach for logical knowledge bases, (ii) the inclusion of large volumes of distributional semantics commonsense knowledge into the inference process and (iii) the provision of a principled geometric representation of the inference process.

This work is organized as follows: section 2 describes a motivational scenario; section 3 provides a brief introduction to distributional semantics ; section 4 describes the logic model; section 5 describes the geometric model; section 6 connects the logical and geometrical models ; section 7 shows the combined distributional-logic inference process; section 8 presents a prototype of the proposed approach; section 9 describes related work and section 10 presents the conclusions and future work.

2 Motivational Scenario

Every knowledge or information artifact (from unstructured text to structured knowledge bases) maps to an implicit or explicit set of user intents and semantic context patterns. The multiplicity of contexts where open domain and commonsense knowledge bases can be used, defines the intrinsic semantic heterogeneity for these scenarios. Different levels of conceptual abstraction or lexical expressions in the representation of predicates and constants are examples where a semantic/terminological gap can strongly impact the inference process.

In the scenario below an user executes a *vocabulary-independent (ontology-agnostic) query* over a logic program Π. A query is *vocabulary independent* if the user is not aware of the terms and concepts inside Π.

Consider the query *'Is the father in law of Bill Clinton's daughter a politician?'* that can be represented as the logical query:

$$? - \mathit{daughter\_of}(X, \mathit{bill\_clinton}), \mathit{politician}(Y), \mathit{father\_in\_law}(Y, X)$$

Let us assume that the logic program Π contains facts and rules such as:

child_of(chelsea_clinton,bill_clinton).
child_of(marc_mezvinsky,edward_mezvinsky).
spouse(chelsea_clinton,marc_mezvinsky).
is_a_congressman(edward_ mezvinsky).
father_in_law(A,B) $\leftarrow$ *spouse(B,C), child_of(C,A).*

meaning that Chelsea is the child of Bill Clinton, Marc Mezvinsky is the child of Edward Mezvinsky, Chelsea is the spouse of Marc, Edward Mezvinsky is a congressman and A is father in law of B when the spouse of B is a child of A.

The inference over Π will not materialize the answer $X = \mathit{chelsea\_clinton}$ and $Y = \mathit{edward\_mezvinsky}$, because despite the statement and the rule describing the same sub-domain, there is no precise vocabulary matching between the query and Π.

In order for the reasoning to work, the approximation of the following terms would need to be established: $\mathit{daughter\_of} \sim \mathit{child\_of}$, $\mathit{is\_a\_congressman} \sim \mathit{politician}$. The reasoner should be able to semantically approximate vocabulary terms such as *daughter_of* and *child_of*, addressing the terminological gap required by this inference.

To close the semantic/vocabulary gap in a traditional deductive logic knowledge base it would be necessary to increase the size of Π to such an extent that it would contain all the facts and rules necessary to cope with any potential vocabulary difference. Together with the aggravation of the scalability problem, it would be necessary to provide a principled mechanism to build such a large scale and consistent set of facts and rules.

3 Distributional Semantics and Semantic Approximative Inference

In this work *distributional semantics* supports the definition of an approximative semantic interpretation for facts and rules in a logic program Π where constants and predicates are mapped to vectors in a *distributional vector space*. This section provides a brief introduction to distributional semantics and outlines the core principles and the rationale of the proposed approximative inference model.

3.1 Distributional Semantics

Distributional semantics is defined upon the assumption that the context surrounding a given word in a text provides important information about its meaning [17]. It focuses on the construction of a semantic model for a word based

on the statistical distribution of co-located words in texts. These semantic models are naturally represented by Vector Space Models (VSMs) [17], where the meaning of a word can be defined by a weighted vector over a term space, which represents the association patterns of co-occurring words in a corpus.

The existence of large amounts of unstructured text on the Web brings the potential to create comprehensive distributional semantic models (DSMs). DSMs can be automatically built from large corpora, not requiring manual intervention on the creation of the semantic model. Additionally, its natural association with VSMs, where dimensional reduction approaches or data structures such as inverted list indexes, can provide a scalability benefit for the instantiation of these models.

These models can provide a more scalable solution to the problem of capturing commonsense semantic information, complementing existing manually created knowledge bases such as Cyc[1]. The computation of semantic relatedness measures between pairs of words is one instance in which the strength of distributional models and methods is empirically supported ([10],[6]).

3.2 The Distributional Inference Vector Space

The commonsense semantic knowledge embedded in a distributional model is used to semantically complement a logic program Π. In a traditional deductive system the inference process is defined by a sequence of exact substitution operations, where the symbols representing constants and predicates are exactly matched under the syntax of the representation language. In the proposed inference model the symbols have an associated *concept vector* representation which encodes its relation to other symbols based on the symbols' co-occurrence statistics in a large unstructured reference corpus. The concept vectors define a distributional vector space which can be used to represent and embed the logic program symbols in the space.

The embedding of logic programs in the distributional vector space allows the definition of a geometric interpretation for the inference process. The geometry allows the definition of a semantic heuristics which defines a direction for the exploration of the solution space.

The proposed inference model uses the lexical-semantic information embedded in a distributional-relational vector space (named τ-Space [7]) to compute a measure of semantic relatedness between logic program symbols in the space. The distributional semantic relatedness measure can be used to establish an approximate semantic equivalence between two predicates at a given context. The intuition behind this approach is that *two terms which are highly semantically related in a distributional model are likely to have a close (implicit) relation*[2].

This work expands on the existing abstraction of the τ-Space, defined in [7], introducing the notion of inference process over a τ-Space, articulating the connections between logical inference and the geometry defined by the τ-Space.

[1] `http://www.cyc.com/platform/opencyc`

[2] Distributional semantic models can be specialized to exclude certain types of semantic relatedness (such as antonyms or relations in a negation context).

4 Logic Model

4.1 Syntax

An alphabet $\mathcal{A}$ is formed by the following disjoint set of symbols: (i) *Predicates* which are represented by $P = \{p_1, \cdots p_m\}$; (ii) *Constants* which are represented by $E = \{e_1, \cdots, e_n\}$; (iii) *Variables* which are represented by upper case letters $\{X, Y, Z, \cdots\}$. Also, we have a fixed set of *connectives* represented by $\{',', \leftarrow, \neg\}$. A *term* t is either a constant or a variable. An *atom* at is an expression of the form $p(t_1, \cdots, t_n)$ where p is a predicate and $t_1, \cdots, t_n$ are terms. An atom $p(t_1, \cdots, t_n)$ is *grounded* whenever $t_1, \cdots, t_n$ are all constants. A (grounded) literal is an (grounded) atom or a negated (grounded) atom.

A *clause* cl is an expression of the form: $head(cl) \leftarrow body(cl).$, where $head(cl)$ is an atom and $body(cl)$ is a conjunction of literals. A *grounded clause* is formed only by grounded literals.

A *logic program* Π is a set of clauses. We say that Π is a definite logic program when there is no negative atom in $body(cl)$. Otherwise, Π is a normal logic program. A *query* Q to Π is an expression of the form $?- q_1, \cdots, q_n.$ where $q_1, \cdots, q_n$ are literals. A *signature* Σ_x is a pair (P_x, E_x) where $P_x \subseteq P$ and $E_x \subseteq E$ are respectively the sets of all predicates and constants that appear in $x \in \{\Pi, at, Q\}$.

4.2 Semantics

Given the sets of constants E and predicates P, let $HU = E$ be the *Herbrand Universe* and HB be the *Herbrand Base* formed by all ground atoms that can be constructed using predicates and constants in P and E.

A Herbrand interpretation of a predicate p is any set $\Im(p) \subseteq HB$ such that all elements in $\Im(p)$ are of the form $p(e_1, \cdots, e_n)$, where for all $i \in [1, n], e_i \in HU$. A Herbrand interpretation $\Im$ *satisfies* a clause cl of the form $h \leftarrow b_1, \cdots, b_n$, $\neg b_{n+1}, \cdots, \neg b_m$ if $h \in \Im$ whenever each $b_1, \cdots, b_n \in \Im$ and each $b_{n+1}, \cdots$, $b_m \notin \Im$. A *Herbrand model* $\mathcal{M}(\Pi) = \bigcup_{p \in P_\Pi} \Im(p)$ of Π is a Herbrand interpretation that satisfies all clauses in Π. A Herbrand model $\mathcal{M}(\Pi)$ is *minimal* if no proper subset of $\mathcal{M}(\Pi)$ is also a model. A definite logic program Π has only one minimal Herbrand model, which we denote as $\mathcal{M}in(\Pi)$.

A set of atoms S is an answer set model of a normal logic program Π iff $S = \mathcal{M}in(\Pi^S)$ where Π^S is the definite logic program obtained from Π (the reduct of Π - [11]): (i) deleting all clauses that has $\neg at$ in its body such that $at \in S$ and (ii) deleting all negated atoms in the bodies of the remaining clauses. An answer set model S satisfies an atom at (resp., $\neg at$) when $at \in S$ (resp., $at \notin S$) which is denoted by $S \models at$ (resp., $S \models \neg at$).

5 Geometrical Model

5.1 τ-Space

The τ-*Space* [7] is a distributional structured vector space model that will be used to represent predicates and constants under a distributional semantic model. It

is built from a *reference corpus* formed by a pair of sets (*Term*, *Context*) where $Term = \{k_1, \cdots, k_t\}$ is a set of terms and $Context = \{c_1, \cdots, c_t\}$ is a set of context windows in the corpus. For example, a given set of documents can be seen as a set of context windows and all terms that occur in those documents form the set of terms.

Term is used to define the basis $Term_{basis} = \{\overrightarrow{\mathbf{k}}_1, \cdots, \overrightarrow{\mathbf{k}}_t\}$ of unit vectors that spans the *term vector space* VS^{term}. In VS^{term}, a context window c_j is represented as:

$$\overrightarrow{\mathbf{c}}_j = \sum_{i=1}^{t} v_{i,j} \overrightarrow{\mathbf{k}}_i \tag{1}$$

where $v_{i,j}$ is 1 if term k_i appears in context window c_j and 0 otherwise.

The set *Context* is used to define the basis $Context_{basis} = \{\overrightarrow{\mathbf{c}}_1, \cdots, \overrightarrow{\mathbf{c}}_t\}$ of vectors that spans the *distributional vector space* VS^{dist}. A term x is represented in VS^{dist} as:

$$\overrightarrow{\mathbf{x}} = \sum_{j=1}^{t} w_j \overrightarrow{\mathbf{c}}_j \tag{2}$$

where

$$w_j = tf_j \times idf = \frac{freq_j}{count(c_j)} \times \log \frac{N}{n_{c_j}} \tag{3}$$

meaning that w_j is the product of the normalized term frequency tf_j (where $freq_j$ is the frequency of term x in the context window c_j and $count(c_j)$ is the number of terms inside c_j) and the inverse document frequency *idf* for the term x (where N is the total number of context windows in the reference corpus and n_{c_j} is the number of context containing the term x).

As consequence, a vector $\overrightarrow{\mathbf{x}} \in VS^{dist}$ can be mapped to VS^{term} by the following transformation:

$$\overrightarrow{\mathbf{x}} = \sum_{i=1}^{t} \sum_{j=1}^{t} w_j v_{i,j} \overrightarrow{\mathbf{k}}_i \tag{4}$$

We can see from the equations above that the set $C \subseteq Context$ where a term occurs defines the concept vectors associated with the term. This represents its meaning on the reference corpus. Since each concept vector is weighted according to the term distribution in the corpus, we can define the set $Context_{basis}$ in terms of $Term_{basis}$ where each dimension maps to a word in the corpus.

6 Linking the Logical and Geometrical Models

In this section, we will define the link between the geometrical (distributional) and logical models. The idea is that the former could provide a way to enrich the semantics and inference power of the latter, resulting in an approach that supports an *approximative semantic matching inference* process.

6.1 Mapping Predicates and Constants to Vectors

The signature of a given logic program Π can be translated into τ-Space vectors in the distributional vector space VS^{dist} as follows:

Definition 1. *Let* $\{\overrightarrow{\mathbf{c}}_1, \cdots, \overrightarrow{\mathbf{c}}_t\}$ *be the vectors basis that spans* VS^{dist}*. The vector representations of* P *and* E *in* VS^{dist} *are defined by:*

$$\overrightarrow{\mathbf{P}}_{VS^{dist}} = \{\overrightarrow{\mathbf{p}} : \overrightarrow{\mathbf{p}} = \sum_{i=1}^{t} v_i^p \overrightarrow{\mathbf{c}}_i, \text{ for each } p \in P\} \tag{5}$$

$$\overrightarrow{\mathbf{E}}_{VS^{dist}} = \{\overrightarrow{\mathbf{e}} : \overrightarrow{\mathbf{e}} = \sum_{i=1}^{t} v_i^e \overrightarrow{\mathbf{c}}_i, \text{ for each } e \in E\} \tag{6}$$

where v_i^e *and* v_i^p *are defined by the weighting scheme over the distributional model. The weighting scheme will reflect the word co-occurrence pattern in the reference corpus.*

Elements of a query Q with signature $\Sigma_Q = (P_Q, E_Q)$ are mapped to VS^{dist} in a similar way.

6.2 Semantic Relatedness of Predicates, Programs and Models

Consider, for example, two highly semantic related concepts represented by two syntactically different predicates, such as $daughter\_of$ and $child\_of$. In the unification process only syntactically identical predicates can be resolved. If we have stated facts/rules using the predicate $child\_of$, no query using predicate $daughter\_of$ would be answered.

In order to bring to logic programs the ability of semantically relate predicate symbols which use a meaningful natural language descriptor, we will define the notion of semantic relatedness between predicates as follows:

Definition 2. *Let* p_1 *and* p_2 *be predicate symbols with same arity and with normalized vector representations* $\overrightarrow{\mathbf{p_1}}$ *and* $\overrightarrow{\mathbf{p_2}}$ *in* VS^{dist}*. The semantic relatedness function* $sr : P \times P \to [0, 1]$ *is defined by the inner product between* $\overrightarrow{\mathbf{p_1}}$ *and* $\overrightarrow{\mathbf{p_2}}$ *:* $sr(p_1, p_2) = \overrightarrow{\mathbf{p_1}}.\overrightarrow{\mathbf{p_2}} = \cos(\theta)$ *where* θ *is the angle between vectors* $\overrightarrow{\mathbf{p_1}}$ *and* $\overrightarrow{\mathbf{p_2}}$*.*

Definition 3. *Let* p_1 *and* p_2 *be predicates and* $\eta \in [0, 1]$ *be a threshold. We say that* p_1 *and* p_2 *are semantically related wrt* η *whenever* $sr(p_1, p_2) > \eta$*.*

The function sr allows us to extend the notion of semantic relatedness to logic programs, answer set models and unification procedure allowing the inference process to continue in cases where predicates are syntactically distinct. Initially, we use the semantic relatedness between predicate symbols to define the predicate substitution as follows:

Definition 4. *Let* $P_1 = \{p_1, \cdots, p_n\}$ *and* $P_2 = \{p'_1, \cdots, p'_n\}$ *be two sets of predicate symbols such that* $\forall i \in [1, n], sr(p_i, p'_i) > \eta$*. A predicate substitution of* P_1 *by* P_2 *wrt* η *is defined by* $\lambda_\eta(P_1, P_2) = \{p_1/p'_1, \cdots, p_n/p'_n\}$*. We denote* $\lambda_\eta^{-1}(P_1, P_2) = \lambda_\eta(P_2, P_1) = \{p'_1/p_1, \cdots, p'_n/p_n\}$*.*

Since the goal of this type of substitution is to allow that the inference process can continue despite the vocabulary differences, definition 4 does not allow the substitution of two different predicates p_i and p_j with a single predicate p'. This is done to preserve the logical semantics of the predicates, that is, both extensions of p_i and p_j. Otherwise, if p' could replace both p_i and p_j, we would have $\Im(p') = \{(c_1, \cdots, c_n)$ such that $(c_1, \cdots, c_n) \in (\Im(p_i) \cup \Im(p_j))\}$.

We associate to predicate substitutions a semantic relatedness measure:

$$sr_{subst}(\lambda_\eta(P_1, P_2)) = \frac{1}{n} * \sum_{i \in [1,n]} sr(p_i, p'_i) \tag{7}$$

which will be also used to define semantic relatedness between logic programs and Herbrand models.

Definition 5. *Let Π_1 and Π_2 be logic programs with signatures, respc., $\Sigma_{\Pi_1} = (P_{\Pi_1}, E_{\Pi_1})$ and $\Sigma_{\Pi_2} = (P_{\Pi_2}, E_{\Pi_2})$. We say that Π_1 and Π_2 are semantically related wrt a threshold η (or sr-logic programs wrt η) when there is some predicate substitution $\lambda_\eta(P_1, P_2)$ such that $\Pi_2 = \Pi_1 \cdot \lambda_\eta(P_1, P_2)$ where $P_1 = (P_{\Pi_1} \setminus P_{\Pi_2})$ and $P_2 = (P_{\Pi_2} \setminus P_{\Pi_1})$.*

Note that when $\Pi_2 = \Pi_1 \cdot \lambda_\eta(P_1, P_2)$, $\Pi_1 = \Pi_2 \cdot \lambda_\eta^{-1}(P_1, P_2) = \Pi_2 \cdot \lambda_\eta(P_2, P_1)$.

Definition 5 states that two sr-logic programs are different versions of the same program that use a set of different predicate symbols, which are semantically related from a natural language perspective. From the logical point of view, the answer set models of Π_1 are preserved in Π_2 (and vice-versa) in the sense that the extensions of all predicates in both programs are the same: different predicate symbols that are semantically related have the same extension. This can be shown as follows:

Proposition 1. *Let Π be a normal logic program, $S \subseteq HB_\Pi$ be a set of atoms. For any predicate substitution λ_η, $(\Pi^S \cdot \lambda_\eta) = (\Pi \cdot \lambda_\eta)^{S \cdot \lambda_\eta}$.*

Proof. ($\subseteq$): Suppose that $cl \in (\Pi^S \cdot \lambda_\eta)$ and $cl \notin (\Pi \cdot \lambda_\eta)^{S \cdot \lambda_\eta}$. Since $cl \in (\Pi^S \cdot \lambda_\eta)$, we have that $cl \cdot \lambda_\eta^{-1} \in \Pi^S$. One of the following cases can occur:

- $cl \cdot \lambda_\eta^{-1} \in \Pi$ when there is no occurrence of negative atoms in the body of cl. Then $cl \in \Pi \cdot \lambda_\eta$ and $cl \in (\Pi \cdot \lambda_\eta)^{S \cdot \lambda_\eta}$, a contradiction; or
- there is a clause $cl' \cdot \lambda_\eta^{-1} \in \Pi$ with negative atoms $\neg at_1 \cdot \lambda_\eta^{-1}, \cdots, \neg at_n \cdot \lambda_\eta^{-1}$ in the body that are all eliminated by S generating $cl \cdot \lambda_\eta^{-1}$. Since $\{at_1 \cdot \lambda_\eta^{-1}, \cdots, at_n \cdot \lambda_\eta^{-1}\} \subseteq S$, we have $\{at_1, \cdots, at_n\} \subseteq (S \cdot \lambda_\eta)$. So $cl \in (\Pi \cdot \lambda_\eta)^{S \cdot \lambda_\eta}$, which contradicts our hypothesis.

($\supseteq$): Suppose that $cl \in (\Pi \cdot \lambda_\eta)^{S \cdot \lambda_\eta}$ and $cl \notin (\Pi^S \cdot \lambda_\eta)$. Since $cl \in (\Pi \cdot \lambda_\eta)^{S \cdot \lambda_\eta}$, we can have one of the following cases:

- $cl \in (\Pi \cdot \lambda_\eta)$. Then $(cl \cdot \lambda_\eta^{-1}) \in \Pi$, and consequently $(cl \cdot \lambda_\eta^{-1}) \in \Pi^S$ or $cl \in (\Pi^S \cdot \lambda_\eta)$, contradicting our hypothesis; or

– there is a clause $cl' \in (\Pi \cdot \lambda_\eta)$ with negative atoms $\neg at_1 \cdot \lambda_\eta, \cdots, \neg at_n \cdot \lambda_\eta$ in the body that are all eliminated by $(S \cdot \lambda_\eta)$ generating cl. Hence $\{at_1, \cdots, at_n\} \subseteq S$ and since $(cl' \cdot \lambda_\eta^{-1}) \in \Pi$, we have that $(cl \cdot \lambda_\eta^{-1}) \in \Pi^S$, or, $cl \in (\Pi^S \cdot \lambda_\eta)$, contradicting our hypothesis.

Corollary 1. *Let Π_1 and Π_2 be sr-logic programs wrt η and S a set of atoms such that $P_S \subseteq P_{\Pi_1}$. Then $\Pi_1^S = (\Pi_2^{S \cdot \lambda_\eta(P_1,P_2)}) \cdot \lambda_\eta(P_2, P_1)$.*

Proof. Since Π_1 and Π_2 be sr-logic programs wrt η, by definition 5, we have $\Pi_1 = \Pi_2 \cdot \lambda_\eta(P_2, P_1)$. Given a set of atoms S, let $S' = S \cdot \lambda_\eta(P_1, P_2)$ and consequently $S = S' \cdot \lambda_\eta(P_2, P_1)$.

Then, we have: $\Pi_1^S = (\Pi_2 \cdot \lambda_\eta(P_2, P_1))^S = (\Pi_2 \cdot \lambda_\eta(P_2, P_1))^{S' \cdot \lambda_\eta(P_2,P_1)} = (\Pi_2^{S'}) \cdot \lambda_\eta(P_2, P_1) = (\Pi_2^{S \cdot \lambda_\eta(P_1,P_2)}) \cdot \lambda_\eta(P_2, P_1)$

Proposition 2. *Let Π_1 and Π_2 be sr-logic programs wrt η.*

$\mathcal{M}(\Pi_1)$ is an answer set model of Π_1 iff $\mathcal{M}(\Pi_2) = \mathcal{M}(\Pi_1) \cdot \lambda_\eta(P_1, P_2)$ is an answer set model of Π_2.

Proof. We have $\Pi_2 = \Pi_1 \cdot \lambda_\eta(P_1, P_2)$ and $\mathcal{M}(\Pi_1) = \mathcal{M}in(\Pi_1^{\mathcal{M}(\Pi_1)})$. So,

$$\mathcal{M}in(\Pi_2^{\mathcal{M}(\Pi_1) \cdot \lambda_\eta(P_1,P_2)}) = \mathcal{M}in((\Pi_1 \cdot \lambda_\eta(P_1, P_2))^{\mathcal{M}(\Pi_1) \cdot \lambda_\eta(P_1,P_2)}) =$$

$$= \mathcal{M}in((\Pi_1^{\mathcal{M}(\Pi_1)}) \cdot \lambda_\eta(P_1, P_2)) = \mathcal{M}in(\Pi_1^{\mathcal{M}(\Pi_1)}) \cdot \lambda_\eta(P_1, P_2) = \mathcal{M}(\Pi_1) \cdot \lambda_\eta(P_1, P_2)$$

The semantic relatedness sr_{prog} between logic programs Π_1 and Π_2 and the semantic relatedness sr_{models} between (answer set) models $\mathcal{M}(\Pi_1)$ and $\mathcal{M}(\Pi_2) = \mathcal{M}(\Pi_1) \cdot \lambda_\eta(P_1, P_2)$ are defined using the predicate substitution $\lambda_\eta(P_1, P_2)$ used to transform Π_1 in Π_2: $sr_{prog}(\Pi_1, \Pi_2) = sr_{models}(\mathcal{M}(\Pi_1), \mathcal{M}(\Pi_2)) = sr_{subst}(\lambda_\eta(P_1, P_2))$

The satisfiability of atoms expressed using a predicate symbol that does not belong to the signature of an answer set model is defined by:

Definition 6. *Let S be an answer set model of a logic program Π. Given a grounded atom $p(t_1, \cdots, t_n)$ such that $p \notin P_\Pi$, we say that:*

– *S sr-satisfies $(p(t_1, \cdots, t_n), \zeta)$ wrt η, denoted by $S \models_\eta (p(t_1, \cdots, t_n), \zeta)$ when there is a substitution $\lambda_\eta(\{p\}, \{p'\})$ for some $p' \in P_\Pi$ such that $S \models (p(t_1, \cdots, t_n) \cdot \lambda_\eta(\{p\}, \{p'\}))$ and ζ is the semantic relatedness measure associated to the predicate substitution $\lambda_\eta(\{p\}, \{p'\})$ as defined in equation (7) (i.e., $\zeta = sr_{subst}(\lambda_\eta(\{p\}, \{p'\}))$).*

– *S sr-satisfies $(\neg p(t_1, \cdots, t_n), \zeta)$ wrt η, denoted by $S \models_\eta (\neg p(t_1, \cdots, t_n), \zeta)$ when there is a substitution $\lambda_\eta(\{p\}, \{p'\})$ for some $p' \in P_\Pi$ such that $S \models (\neg p(t_1, \cdots, t_n) \cdot \lambda_\eta(\{p\}, \{p'\}))$ and ζ is the semantic relatedness measure associated to the predicate substitution $\lambda_\eta(\{p\}, \{p'\})$ as defined in equation (7) (i.e., $\zeta = sr_{subst}(\lambda_\eta(\{p\}, \{p'\}))$).*

Given a set of grounded literals Q such that $Q = Q_1 \cup Q_2$, $P_{Q_1} \cap P_{Q_2} = \emptyset$, $P_{Q_1} \subseteq P_\Pi$ and $P_{Q_2} \not\subseteq P_\Pi$, we say that

– *S sr-satisfies (Q, ζ) wrt η, denoted by $S \models_\eta (Q, \zeta)$ iff there is a substitution $\lambda_\eta(P_{Q_2}, P')$ for some $P' \subseteq P_\Pi$ such that $S \models (Q \cdot \lambda_\eta(P_{Q_2}, P'))$ and ζ is the semantic relatedness measure associated to the predicate substitution $\lambda_\eta(P_{Q_2}, P')$ as defined in equation (7) (i.e., $\zeta = sr_{subst}(\lambda_\eta(P_{Q_2}, P'))$).*

7 Distributional-Logic Inference

In this section, we will present the combined distributional-logical inference process. The first step to answer Q is to order the literals in it according to a relevance order of elements in $P_Q \cup E_Q$.

Definition 7. *Let Q be a query. The relevance order of the literals in Q is the sequence of literals $< l_1, l_2, \cdots, l_m >$ such that: (i) Q is equivalent to $\bigwedge_{i=1}^{m} l_i$; (ii) $\forall i \in [1, m-1], f_{relevance}(l_i) \geq f_{relevance}(l_{i+1})$*

The function $f_{relevance}$ is a heuristic measure of specificity over the query symbols which gets the most specific constant or predicate (which we call *semantic pivot symbol*). The specificity can be defined as the IDF (Inverse Document Frequency) of a term over a reference corpus, as a function of the *lexical categories* associated with the term, or as a combination of the number of elements associated with x (where x is a predicate or constant). The rationale behind prioritizing the selection of a symbol with high specificity is that the algorithm prioritizes the hardest constraint in the query and selects the query element less prone to semantic ambiguity, vagueness and polysemy. Normally the first semantic pivot symbol selected in a query Q is a constant, if any exists.

The selection of a semantic pivot allows a reduction in the search space where just the elements of Π associated with the pivot at a given iteration are candidates for the semantic matching. In each iteration, a set of semantic pivots is defined, which propagates to other points in the τ-Space, following the topological relations defined by the syntactic structure of the atoms. The order of the sequence is unique, with regard to a $f_{relevance}$ function and the syntactic constrains of the query elements.

Once the order of literals in a query Q is fixed, to answer the ordered query Q over Π, first we use algorithm 1 to find all predicate substitutions wrt a given threshold η between the predicate symbols that appear in Π (P_Π) and all the predicate symbols q in Q such that $q \notin P_\Pi$ (P_{query}).

Each predicate substitution $\lambda_\eta(P_{query}, P'_\Pi)$ generated by algorithm 1 can be applied to Q resulting in a query $(Q \cdot \lambda_\eta(P_{query}, P'_\Pi))$ where all predicates belong to P_Π. So, we can answer this transformed query using any answer set solver.

Note that for each $\lambda_\eta(P_{query}, P'_\Pi) \in$ *Substitutions* we can calculate the score associated with that substitution $(sr_{subst}(\lambda_\eta(P_{query}, P'_\Pi)))$ since the semantic relatedness measure sr is stored whenever a substitution is found (line 13 in algorithm 1).

Algorithm 1. Distributional Predicate Substitution Algorithm - DPS

INPUT

- P_{Π}: The list of all predicate symbols that appear in a program Π
- P_{query}: The list of all predicate symbols q that appear in a query Q such that $q \notin P_{\Pi}$
- η: Threshold

OUTPUT

- *Substitutions*: A set with all predicate substitutions $\lambda_{\eta}(P_{query}, P'_{\Pi})$ where $P'_{\Pi} \subseteq P_{\Pi}$ and $|P'_{\Pi}| = |P_{query}|$

PROCEDURE $DPS(P_{\Pi},P_{query},\eta)$:

```
1: if P_query == [ ] then
2:    return  ([ [ ] ])
3: else
4:    for all i ∈ [1, |P_query|] do
5:       X ← P_query(i)
6:       P'_query ← remove(X, P_query)
7:       Substitutions ← [ ]
8:       for all Y ∈ P_Π do
9:          if sr(X, Y) > η then
10:            P'_Π ← remove(Y, P_Π)
11:            Subst ← [ ]
12:            for all Z ∈ DPS(P'_Π,P'_query,η) do
13:               Subst ← append(Z, [(X, Y, sr(X, Y))])
14:               Substitutions ← append(Substitution, [Subst])
15:            end for
16:         end if
17:      end for
18:   end for
19: end if
20: return  Substitutions
```

8 Prototype and Evaluation

A prototype of the proposed approach was built and it contains two modules: (i) the prolog module implemented using SWI-Prolog[3] which identifies if a predicate in a query belongs or not to the signature of a given normal logic program and does all predicate substitutions with the respective semantic relatedness measure; (ii) the τ-Space module, which was constructed using Explicit Semantic Analysis (ESA) as the distributional model built over Wikipedia 2006, where the Wikipedia articles were the context windows and TF/IDF was the weighting scheme.

The query is of the form (Q, η), where Q is a query and η is the desired threshold which has its value determined experimentally accordingly to the cor-

[3] `www.swi-prolog.org/`

pus that is used. The experimental threshold η was based on the semantic differential approach for ESA proposed in [6]. The approach was simplified to a ground threshold of 0.05.

The answers to (Q, η) are usual logic program answers with the scores corresponding to sr_{subst}. When the predicate q in the selected literal l of Q is identified as not belonging to P_Π, the τ-Space module is called and returns all predicate names of Π semantically related to q wrt η. Each one of these predicates (if any exists) replaces q in the query, which proceeds in the inference process as usual.

Example 1. Let Π be formed by:

child_of(chelsea_clinton, bill_clinton).
child_of(marc_mezvinsky,edward_mezvinsky).
spouse(chelsea_clinton, marc_mezvinsky).
is_a_congressman(edward_mezvinsky).
father_in_law(A,B) ← spouse(B,C),child_of(C,A).

Suppose that we want to answer the query *"Is the father in law of Bill Clinton's daughter a politician?"* wrt a threshold $\eta = 0.05$:

?-((daughter_of(X,bill_clinton),father_in_law(Y,X),politician(Y)),0.05).

Since the predicate *daughter_of* does not appear in Σ_Π, we need to verify if there is a semantically related binary predicate to *daughter_of* wrt $\eta = 0.05$. As can be seen in table 1, only *child_of* is semantically related to *daughter_of* wrt η ($sr(child\_of, daughter\_of) = 0.054 > 0.05$). Thus, we allow that these predicates unify and they have a *mgu* ($\{X/chelsea\_clinton\}, 0.054$). The complete inference is shown in figure 1 and the score of the answer is $(0.054 + 0.06)/2 = 0.057$.

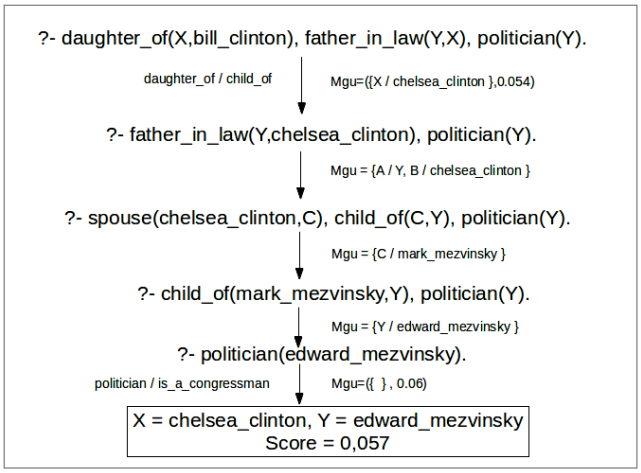

Fig. 1. Derivation for the question *"Is the father in law of Bill Clinton's daughter a politician?"*

Table 1. Semantic relatedness determined by the τ-Space module between the predicates in Q and Π, according to arity

sr	*child_of*/2	*spouse*/2	*father_in_law*/2	*is_a_congressman*/1
daughter_of/2	0.054	0.012	0.048	-
politician/1	-	-	-	0.06

As an approximative approach, we can have some undesirable answers as shows the following example:

Example 2. Suppose that we have Π' defined as:

$$\Pi \cup \{spouse(bill\_clinton, hilary\_clinton), child\_of(hilary\_clinton, hugh\_rodman)\}$$

where Π is the program defined in example 1.

Consider that we query Π' with *"Who is Bill Clinton's daughter ?"* using a threshold $\eta = 0.04$. In this case, we have two predicates semantically related to *daughter_of* obtaining the answers:

- X = *chelsea_clinton* with score 0.054, replacing *daughter_of* by *child_of*,
- X = *hugh_rodman* with score 0.048, replacing *daughter_of* by *father_in_law*.

These answers could be filtered using a higher threshold as input (in the example, 0.05), using a threshold over the final score or through a principled interaction mechanism (dialog/disambiguation system).

To illustrate the use of negation, we present the following example:

Example 3. Suppose we query Π' using $\eta = 0.04$ with:

?- ($\neg$ *daughter_of(chelsea_clinton,bill_clinton),0.04).*

We obtain the following answers:

- **no** with score 0.054, replacing $daughter\_of$ by $child\_of$ (since $child\_of(chelsea\_clinton, bill\_clinton) \in \Pi'$),
- **yes** with score 0.048, replacing $daughter\_of$ by $father\_in\_law$ (since $child\_of(hilary\_clinton, chelsea\_clinton) \notin \Pi'$).

As before, the answers could be filtered either manually or by adjusting the threshold.

To evaluate the semantic matching (τ-Space module) we used DBpedia[4], a heterogeneous and large-scale data set which consists of 45,767 predicates, 5,556,492 classes, 9,434,677 instances, as knowledge base. The relevance function used a combination of IDF over predicates, cardinality (number of associated constants to another constant) and a dice string similarity coefficient.

[4] `http://dbpedia.org`

To query this knowledge base we selected 18 queries (Table 2) extracted from the Question Answering over Linked Data (QALD)[5] 2011 test collection. The selected subset concentrates on queries with a vocabulary gap between query and knowledge base terms. Queries in the original test collection with a perfect vocabulary match and with functional operators were removed.

The approach achieved **avg. recall=0.935**, corroborating the hypothesis that distributional semantics provides a comprehensive semantic matching solution. **Average mean reciprocal rank (mrr) = 0.632** shows that most of the results are in the first two positions of the ranked list. Different from traditional approaches where the matching is done at a syntactical level, the semantic approximation implies that absolute precision will unlikely to be achieved in all cases since some level of ambiguity and vagueness is intrinsic to the vocabulary gap problem.

The **avg. precision=0.561** confirms that the distributional semantic relatedness measure is able to provide a selective semantic filtering mechanism. However, in the context of logic programs higher precision should be targeted by the use of more selective distributional models and by the introduction of disambiguation/dialog user feedback mechanisms. ESA is a distributional model which, by its construction, favours the broader class of semantic relatedness instead of the more constrained class of semantic similarity (such as taxonomic relations). The use of ESA favours recall and broader vocabulary independency over precision and assume that noisy inferences can be filtered out by a disambiguation mechanism. A relevant research direction is to imp[illegible] precision by using distributional models with narrower context windows [??]. Enlarging the set of inferences can be problematic in large-scale knowledge bases, as for example, in the context of the Semantic Web. The composition with scalable and selective reasoning models (e.g. in Bonatti et al. [4]) should be investigated in order to minimize the impact of the additional inference process.

The average predicate distributional **matching time is 1,523 ms** in a core i5 8GB RAM machine. The τ-Space works as a *semantic best-effort* approximation [7] mechanism where there are no warranties of absolute precision but recall is close to 1. The distributional semantic relatedness measure provides a high selectivity rate over unrelevant results (shown by the precision value). These assumptions mean that in most cases the final result is found, but spurious inferences are present in the current distributional models. These spurious inferences can be eliminated by the provision of dialog mechanisms, where users can provide additional information in order to disambiguate the query.

The computational cost of the distributional semantic approximation concentrates on the cosine similarity operation for the semantic relatedness computation which can be performed at O(nlogn) time complexity using Locality Sensitive Hashing (LSH) techniques.

[5] `www.sc.cit-ec.uni-bielefeld.de/qald-1`

Table 2. Examples of prolog queries in the test collection. In the third column, (p,r,fm) represents (precision, recall, F-measure)

NL-query	**Prolog-query**	(**p**, **r**, **fm**)	**vocabulary gap (query=dataset)**
Who was the wife of Abraham Lincoln?	wife(X,abraham_lincoln)	(0.0305, 1,0.0592)	wife = spouse, President Lincoln = Abraham Lincoln
Who created English Wikipedia ?	created(X,english_wikipedia)	(1,1,1)	created = author, English Wikipedia = English Wikipedia
Who is the owner of Aldi?	owner(X,aldi)	(0.3333, 1, 0.5)	owns = key Person, Aldi = Aldi
How tall is Claudia Schiffer?	tall(claudia_schiffer,X)	(0,09090,1, 0.1667)	Claudia Schiffer = Claudia Schiffer, tall = height
Is Natalie Portman an actress?	actress(natalie_portman)	(1,1,1)	Natalie Portman = Natalie Portman, actress = Actor
Who wrote the book The Pillars of the Earth?	wrote(X,the_pillars_of_earth)	(0.5, 0.5, 0.5)	wrote = author, The Pillars of the Earth = The Pillars of the Earth
Who was Tom Hanks married to?	married_to(X,tom_hanks)	(0.75, 1, 0.8571)	Tom Hanks = Tom Hanks, married to = spouse
When was Lucas Arts founded?	founded(lucas_arts,X)	(1,1,1)	Lucas Arts = Lucas Arts, founded = foundation
Who is the daughter of Bill Clinton married to?	daugther_of(X,bill_clinton), married_to(X,Y)	(0.5, 0.5, 0.5)	Bill Clinton = Bill_Clinton, daughter = child, married to = spouse
Where did Abraham Lincoln die?	die(abraham_lincoln,X)	(0.0162, 1, 0.032)	Abraham Lincoln = Abraham Lincoln, die = death Place
Who is the mayor of New York City ?	mayor(X,new_york_city)	(0.2, 1, 0.3333)	New York City = New York City, mayor = leader Name
What is the profession of Frank Herbert ?	profession(frank_herbert,X)	(0.01428, 1, 0.0281)	Frank Herbert = Frank Herbert, profession = occupation
What did Bruce Carver die from ?	die(bruce_carver,X)	(0.1818, 1, 0.3077)	Bruce Carver = Bruce Carver, die = death Cause
Who designed the Brooklyn Bridge ?	designed(X,brooklyn_ridge)	(0.5, 1, 0.6667)	Brooklyn Bridge = Brooklyn Bridge, designed = designer
Give me all films produced by Hal Roach?	produced(hal_roach,X)	(0.98, 0.9722, 0.9761)	films = Film, produced = producer, Hal Roach = Hal Roach
When was Capcom founded ?	founded(capcom,X)	(1, 1, 1)	Capcom = Capcom, founded = foundation
Which albums contain the song Last Christmas?	contains(X,last_christmas)	(1, 0.8571, 0.9231)	music albums = album, contain = , song = single, Last Christmas = Last Christmas
Was U.S. president Jackson involved in a war ?	u_s_president(jackson), involked(jackson,war)	(1, 1, 1)	U.S. president = Presidents Of The United States, Jackson = Jackson, war = battle

9 Related Work

In [13], Lukasiewicz & Straccia presented probabilistic fuzzy dl-programs, which is a uniform framework that deals with uncertainty and fuzzy vagueness. Our work focus on the ontology mapping aspect (uncertainty) and in the use of a distributional semantic approach to align semantically equivalent terms. The common goal of both fuzzy/probabilistic and distributional approaches is the introduction of flexibility into the reasoning process. The main benefit of using distributional semantics is the use of large-scale unstructured or semi-structured information sources to complement the semantics of logic programs. One of the streghts of distributional semantic models is from the acquisitional perspective, where comprehensive semantic models can be automatically built from large-scale corpora.

Distributional semantic models are evolving in the direction of coping with better compositional principles, supporting the semantic interpretation of complex sentences/statements. Baroni et al. [3] provide an extensive discussion of state of the art approaches for compositional-distributional models. In this work the compositional model is given by the structure of the logical atoms in a logic program Π, which defines a set of vectors in the distributional vector space.

In [12], Grefenstette presented how elements of a quantifier-free predicate calculus can be modelled using tensors and tensor contraction. The basic elements, truth values and domains objects, are modelled as vectors and predicates and relations are modelled through high order tensors. Also, Boolean connectives are modelled using tensors and with the basic elements used to build a quantifier-free predicate calculus.

Research on schema matching/alignment [5] have extensively investigated semantic matching approaches for entities on different schemas. Different matching strategies are employed ranging from structural approaches to strategies based on linguistic resources [5]. Most of the approaches focusing on linguistic resources concentrate on the use of manually created resources such as WordNet. Distributional semantic models are still not extensively used in this context. Another difference between this work and schema alignment approaches is the context in which the semantic matching takes place, which here focuses on the query - knowledge base semantic matching.

Freitas et al. [8] and Novacek et al. [9] describe distributional approaches applied to Semantic Web Data. While Freitas et al. [8] focuses on a natural language query scenario, [9] Novacek et al. targets the description of a tensor-based model for RDF data and its evaluation on entity consolidation.

10 Conclusion and Future Work

This work presented a principled approximative inference model for large-scale and heterogeneous knowledge bases which adds the flexibility and the scale of commonsense-based semantic approximation of distributional semantics to logic programming models. The approach was formalized, a prototype was implemented and evaluated over a large knowledge base, achieving avg. recall=0.935,

avg. mean reciprocal rank=0.632 and avg. precision=0.561. The proposed approach provides a provides a high recall and mean reciprocal rank semantic matching mechanism, under a semantic best-effort scenario (accurate approximation, but which demands a user interaction or post processing step).

Future work will concentrate on the implementation of a pre-processing strategy for natural language queries, the investigation of more constrained distributional models focussing on the improvement of precision and the study of the connection of our approach to synonymous theories in answer set programming proposed by Pearce and Valverde [16].

Acknowledgments. The work presented in this paper has been funded by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2). João C. P. da Silva is a CNPq Fellow - Science without Borders (Brazil).

References

1. Abelson, H., Sussman, G.-J., Sussman, J.: Structure and Interpretation of Computer Programs. MIT Press, Cambridge (1985)
2. Baral, C.: Knowledge Representation, Reasoning, and Declarative Problem Solving. Cambridge University Press, Cambridge (2003)
3. Baroni, M., Bernardi, R., Zamparelli, R.: Frege in Space: A Program for Compositional Distributional Semantics. Linguistic Issues in Language Technologies (to appear)
4. Bonatti, P.A., Hogan, A., Polleres, A., Sauro, L.: Robust and Scalable Linked Data Reasoning Incorporating Provenance and Trust Annotations. Journal of Web Semantics 9(2), 165–201 (2011)
5. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. Journal on Data Semantics 4, 146–171 (2005)
6. Freitas, A., Curry, E., O'Riain, S.: A Distributional Approach for Terminology-Level Semantic Search on the Linked Data Web. In: 27th ACM Symposium On Applied Computing (SAC 2012). ACM Press (2012)
7. Freitas, A., Curry, E., Oliveira, J.G., O'Riain, S.: A Distributional Structured Semantic Space for Querying RDF Graph Data. International Journal of Semantic Computing 5(4), 433–462 (2012)
8. Freitas, A., Oliveira, J.G., Curry, E., O'Riain, S.: A Multidimensional Semantic Space for Data Model Independent Queries over RDF Data. In: Proceedings of the 5th International Conference on Semantic Computing, ICSC (2011)
9. Nováček, V., Handschuh, S., Decker, S.: Getting the Meaning Right: A Complementary Distributional Layer for the Web Semantics. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 504–519. Springer, Heidelberg (2011)
10. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 1606–1611 (2007)
11. Gelfond, M., Lifschitz, V.: Logic programs with classical negation. In: Logic Programming: Proc. of the Seventh International Conf., pp. 579–597 (1990)
12. Grefenstette, E.: Towards a Formal Distributional Semantics: Simulating Logical Calculi with Tensors. CoRR (2013)

13. Lukasiewicz, T., Straccia, U.: Description logic programs under probabilistic uncertainty and fuzzy vagueness. International Journal of Approximate Reasoning 50, 837–853 (2009)
14. Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in description logics for the Semantic Web 6, 291-308 (2008)
15. Lloyd, J.W.: Foundations of Logic Programming. Springer-Verlag New York, Inc., USA (1993)
16. Pearce, D., Valverde, A.: Synonymus Theories in Answer Set Programming and Equilibrium Logic. In: Proc. of 16th Eureopean Conference on Artificial Intelligence (ECAI 2004), pp. 388–392 (2004)
17. Turney, P.D., Pantel, P.: From frequency to meaning: vector space models of semantics. J. Artif. Int. Res. 37(1), 141–188 (2010)