Semantic Web 0 (0) 1
IOS Press

Vecsigrafo: corpus-based word-concept embeddings

Bridging the statistic-symbolic representational gap in natural language processing

Ronald Denaux [a,*] and Jose Manuel Gomez-Perez [a,**]
[a] *Cogito Labs, Expert System, Madrid, Spain*
E-mails: rdenaux@expertsystem.com, jmgomez@expertsystem.com

Abstract. The proliferation of knowledge graphs and recent advances in Artificial Intelligence have raised great expectations related to the combination of symbolic and distributional semantics in cognitive tasks. This is particularly the case of knowledge-based approaches to natural language processing as near-human symbolic understanding relies on expressive, structured knowledge representations. Engineered by humans, such knowledge graphs are frequently well curated and of high quality, but at the same time can be labor-intensive, brittle or biased. The work reported in this paper aims to address such limitations, bringing together bottom-up, corpus-based knowledge and top-down, structured knowledge graphs by capturing as embeddings in a joint space the semantics of both words and concepts from large document corpora. To evaluate our results, we perform the largest and most comprehensive empirical study around this topic that we are aware of, analyzing and comparing the quality of the resulting embeddings over competing approaches. We include a detailed ablation study on the different strategies and components our approach comprises and show that our method outperforms the previous state of the art according to standard benchmarks.

Keywords: joint word and concept embeddings, corpus-based embeddings, embeddings quality, knowledge graphs, analysis and benchmarking

1. Introduction

The history of Artificial Intelligence is a quest for the perfect combination of reasoning performance and the ability to capture knowledge in machine-actionable formats. Early AI systems developed during the '70s like MYCIN [1] already proved it was possible to effectively emulate human reasoning in tasks like classification or diagnosis through artificial means. However, the acquisition of expert knowledge from humans soon proved to be a challenging task, resulting in what was known ever after as the knowledge acquisition bottleneck [2].

Indeed, in an attempt to address this challenge and work at the knowledge level [3], knowledge acquisition eventually became a modeling activity rather than a task focused on extracting knowledge from the mind of an expert. Along the knowledge level path came ontologies, semantic networks and eventually knowledge graphs (henceforth, KGs), which provide rich, expressive and actionable descriptions of the domain of interest and support logical explanations of reasoning outcomes. However, KGs can be costly to produce and scale since a considerable amount of well-trained human labor [4] is needed to manually encode high-quality knowledge in the required formats. Furthermore, the design decisions made by knowledge engineers can also have an impact in terms of depth, breadth and focus, which may result in biased and/or brittle knowledge representations, hence requiring continuous supervision and curation.

In parallel, the last decade has witnessed a noticeable shift from knowledge-based, human-engineered methods to statistical ones due to the increasing availability of raw data and cheaper computing power, facilitating the training of increasingly effective models in areas of AI like natural language processing (NLP).

*Corresponding author. E-mail: rdenaux@expertsystem.com.
**Corresponding author. E-mail: jmgomez@expertsystem.com.

Recent results in the field of distributional semantics e.g. word2vec [5] are particularly promising ways to capture the meaning of words in document corpora as a vector in a dense, low-dimensional space. Among their applications, word embeddings have proved to be useful in term similarity, analogy and relatedness, as well as many NLP downstream tasks including e.g. classification [6], question answering [7–9] or machine translation [10–13]. However, the knowledge thus captured is generally hard to interpret, let alone matched to existing concepts and relations [14], i.e. the main artifacts (explicitly) represented in a KG.

In the intersection of knowledge-based and statistical approaches to NLP, many argue [15–17] that KGs can enhance both expressivity and reasoning power, and advocate for a hybrid approach leveraging the best of both worlds. This is particularly the case in situations where there is a lack of sufficient data or an adequate methodology to learn the nuances associated to the concepts and the relationships between them, which on the other hand can be explicitly (and extensively) represented in existing KGs.

Furthermore, recent contributions [18–21] show evidence that it is possible to learn meaning exclusively from text through NLP tasks such as language modeling and textual entailment, capturing not only lower-level aspects of the text like syntactic information but also showing a certain abstraction capability in the form of context-dependent aspects of word meaning that can be used e.g. for word-sense disambiguation. A different line of research, focused on the pragmatics of language [22, 23], on the contrary argues that effectively capturing meaning requires not only taking into account the form of the text but also other aspects like the particular context in which such form is used or the intent or cognitive state of the speaker, suggesting that the text needs to be enriched in order to actually convey the required meaning to a learner. For NLP to scale beyond partial, task-specific solutions, it must be informed by what is known about how humans use language.

In this paper, we take a step further and argue that the application of structured knowledge contained in KGs can provide such necessary guidance to extract actual meaning from text. In doing so, we produce disambiguated, joint word and concept embeddings that follow a hybrid knowledge formalism involving statistic and symbolic knowledge representations. Also, we argue that the resulting representations are richer and more expressive than those produced through existing approaches based on word-only, KG and hybrid word-concept embeddings.

To support our claims we run a comprehensive set of experiments with different learning algorithms over a selection of document corpora in varying sizes and forms and evaluate our results over several NLP tasks, both intrinsic (semantic similarity, relatedness) and extrinsic (word-concept and relation prediction). We conduct an ablation study[1] that takes into account different disambiguation strategies and variations of our approach, providing a deeper insight on the different aspects our method comprises and illustrating how the quality of the resulting embeddings evolves over the different strategies followed to generate them and the size of the training corpus. We study the effects of filtering over raw text based on grammatical information, entities and other criteria, the effects of (and different approaches to) lemmatization, the impact of jointly training lexical and semantic embeddings, and the effects of applying different disambiguation strategies. Finally, we propose a number of mechanisms to measure the quality and properties of the resulting embeddings, including word and concept prediction plots and inter-embedding agreement. Our results show that our approach consistently outperforms word-only and knowledge graph embeddings, as well as most of the hybrid baselines.

The paper is structured as follows. Next section provides an overview of the research context relevant to our work in areas including word, graph and sense embedding. Section 3 describes our approach to capture as embeddings the semantics of both words and concepts in large document corpora. Section 4 goes on to evaluate our results over different datasets and tasks, comparing to the approaches described in section 2, including a comparative study and an ablation study. Next, section 5 reflects on our findings and provides additional insight and interpretation of the evaluation results. Finally, section 6 concludes the paper and advances next steps in our research.

2. Related work

This work is among the first few to study the joint learning of embeddings for words and concepts from a large disambiguated corpus. The idea itself is not

[1] Systematically removing some of the information fed to our embedding learning system to determine individual contributions to the overall performance.

novel, as Camacho-Colladas et al. [24] points out, but performing a practical study is difficult due to the lack of manually sense-annotated datasets. The largest such dataset is SemCor [25] (version 3.0), a corpus of 537K tokens, 274K of which are annotated with senses from WordNet. Although this dataset could be used with our approach to generate embeddings for the WordNet senses, results from work on word-embeddings show that the size of the corpus greatly affects the quality of the learned embeddings and that corpora in the order of billion tokens are required. In this paper we use an automatic approach for generating word-concept annotations, which makes it possible to use large corpora to learn good quality concept and word embeddings as our studies and results in section 4 show.

Below, we discuss several similar approaches varying from those learning plain word-embeddings, to those learning sense and concept embeddings from corpora and semantic networks, and those which do not use corpora at all, but instead attempt to learn concept embeddings directly from a knowledge graph.

2.1. Word embeddings

Learning word embeddings[2] has a relatively long history [26], with earlier works focused on deriving embeddings from co-occurrence matrices and more recent work focusing on training models to predict words based on their context [27]. Both approaches are roughly equivalent as long as design choices and hyperparameter optimization are taken into account [28].

Most of the recent work in this area was triggered by the word2vec algorithm proposed in [5] which provided an efficient way to learn word embeddings by predicting words based on their context words[3] and using negative sampling. Recent improvements on this family of algorithms [29] also take into account (i) subword information by learning embeddings for 3 to 6 character n-grams, (ii) multi-words by pre-processing the corpus and combining n-grams of words with high mutual information like “New_York_City” and (iii) learning a weighting scheme (rather than predefining it) to give more weight to context words depending on their relative position to the center word[4]. These advances are available via the FastText implementation and pretrained embeddings. Algorithms based on word co-occurrences are also available. GloVe [30] and Swivel [31] are two algorithms which learn embeddings directly from a sparse co-occurrence matrix that can be derived from a corpus; they do this by calculating relational probabilities between words based on their co-occurrence and total counts in the corpus.

These approaches have been shown to learn lexical and semantic relations. However, since they stay at the level of words, they suffer from issues regarding word ambiguity. And since most words are polysemic, the learned embeddings must either try to capture the meaning of the different senses or encode only the meaning of the most frequent sense. In the opposite direction, the resulting embedding space only provides an embedding for each word, which makes it difficult to derive an embedding for the concept based on the various words which can be used to refer to that concept.

The approach described in this paper is an extension that can be applied to both word2vec style algorithms and to co-occurrence algorithms. In this paper we only applied this extension to Swivel, although applying it to GloVe and the standard word2vec implementations should be straightforward. Applying it to FastText would be more complicated, especially when taking into account the sub-word information, since words can be subdivided into character n-grams, but concepts cannot.

2.2. Sense and concept embeddings

A few approaches have been proposed to produce sense and concept embeddings from corpora. One approach to resolve this is to generate *sense embeddings* [32], whereby the corpus is disambiguated using Babelfy and then word2vec is applied over the disambiguated version of the corpus. Since plain word2vec is applied, only vectors for senses are generated. Jointly learning both words and senses was proposed by Chen et al. [33] and Rothe et al. [34] via multi-step approaches where the system first learns word embeddings, then applies disambiguation based on WordNet and then learns the joint embeddings. While this addresses ambiguity of individual words, the resulting embeddings focus on synonymous word-sense pairs[5], rather than on KG concepts.

[2] Also called the Vector Space Model in the literature.

[3] or viceversa, respectively called continuous bag-of-words (cbow) and skip-gram architectures.

[4] Sometimes also called "target" or "focus" word in the literature.

[5] E.g. word-sense pairs $\texttt{apple}^{\texttt{N}}_{\texttt{2}}$ and $\texttt{Malus\_pumila}^{\texttt{N}}_{\texttt{1}}$ have separate embeddings, but the concept for *apple tree* they represent has no embedding.

Another approach for learning embeddings for concepts based on a corpus without requiring word-sense disambiguation is NASARI [24], which uses lexical specificity to learn concept embeddings from Wikipedia subcorpora. These embeddings have as their dimensions, the lexical specificity of words in the subcorpus, hence they are sparse and harder to apply than low-dimensional embeddings such as those produced by word2vec. For this reason, NASARI also proposes to generate "embedded vectors" which are weighted averaged vectors from a conventional word2vec embedding space. This approach only works for Wikipedia and BabelNet, since you need a way to create a subcorpus that is relevant to entities in the knowledge base. Furthermore, although this approach should support concept embeddings for all types of words, the pre-trained embeddings we found only provided embeddings for noun concepts[6].

Finally, the work that is closest to our work is SW2V (Senses and Words to Vectors) [35] which proposes a lightweight word-disambiguation algorithm and extends the Continuous Bag of Words architecture of word2vec to take into account both words and senses. Our approach is essentially the same, although there are various implementational differences: (i) we use our proprietary disambiguator; (ii) we implemented our learning algorithm as a variation of correlation-based algorithms as a consequence; and (iii) we take into account the distance of context words and concepts to the center word. In terms of evaluation, Mancini et al. [35] only reports results for 2 word-similarity datasets while we provide an extensive analysis on 14 datasets. We further analyze the impact of different corpus sizes and look into the inter-agreement between different vector spaces (a measure of how similar two vector spaces are based on the predicted distances between a set of word pairs).

2.2.1. Graph embeddings

Several approaches have been proposed to create concept embeddings directly from knowledge graphs, such as TransE [36], HolE [37], ProjE [38], RDF2Vec [39] and Graph Convolutions [40]. The main goal of such concept embeddings is typically graph completion. In our opinion, these approaches all have the same drawback: they encode the knowledge (including biases) explicitly contained in the source knowledge graph, which is typically already a condensed and filtered version of the real world data. Even large knowledge graphs only provide a fraction of the data that can be gleaned from raw datasets such as Wikipedia and other web-based corpora; i.e. these embeddings cannot learn from raw data as it appears in the real-world. In our evaluation we have used HolE to compare how such word and concept embeddings compare to those derived from large text corpora.

[6] See Section 4.4.1 for a description of and a link to the pre-trained embeddings we used.

3. Corpus-based joint concept-word embeddings

In order to build hybrid systems which can use both bottom-up (corpus-based) embeddings and top-down (KG) knowledge, we propose to generate embeddings which share the same vocabulary as the KGs. This means generating embeddings for knowledge items represented in the KG such as concepts and surface forms (words and expressions) associated to the concepts in the KG[7].

3.1. Notation and preliminaries

Let T be the set of *tokens* that can occur in text after some tokenization is applied; this means tokens may include words ("running"), punctuation marks (";"), multi-word expressions ("United States of America") or combinations of words with punctuation marks ("However,", "–"). Let L be the set of *lemmas*: base forms of words (i.e. without morphological or conjugational variations). Note that $L \subset T$.[8] We also use the term **lexical entry** –or simply **word**– to refer to a token or a lemma. Let C be the set of concept identifiers in some KG, we use the term **semantic entry** –or simply **concept**– to refer to elements in C. Let $V \subset T \cup C$ be the set of lexical and semantic entries for which we want to derive embeddings, also called the **vocabulary**. A corpus is a sequence of tokens $t_i \in T$; we follow and extend the definition of **context** around a token (used in e.g. word2vec, GloVe and Swivel) as a W-sized sliding window over the sequence of tokens. Therefore we say that tokens $t^{i-W}, ..., t^{i-1}, t^{i+1}, ..., t^{i+W}$ are in the context of center token t^i in the context at position i in the corpus. Each

[7] In RDF, this typically means values for `rdfs:label` properties, or words and expressions encoded as `ontolex:LexicalEntry` instances using the lexicon model for ontologies (see https://www.w3.org/2016/05/ontolex/).

[8] We assume lemmatization correctly strips away punctuation marks (e.g. lemma of "However," is "however" and lemma of "Dr." is "Dr.")

context can be represented as a collection of center-context pairs of the form (t_i, t_j), where $t_i \in T$ and $t_j \in T$. We extend this to take into account lemmas and concepts: let D be the collection of center-context entry pairs (x_i, x_j) observed in a corpus, where $x_i \in V$ and $x_j \in V$.[9] We use notation $\#(x_i, x_j)$ to refer to the number of times the center entry x_i co-occurred with context entry x_j in D. We also define $p(x_i, x_j)$ as the set of positions in the corpus where x_i is the center word and x_j is a context word. Similarly $\#(x)$ is the number of times entry x occurred in D as a center word. Finally, let d be the dimension of the vector space, so that for each entry x in V, has two corresponding vectors $\vec{x}_C$ and $\vec{x}_F \in \mathbb{R}^d$, which correspond to the vector representation of x as a context or as a center entry, respectively.

3.2. Formal definition

The overall process for learning joint word and concept embeddings in what we call a Vecsigrafo (derived from the term Sensigrafo, Expert System's KG[10]) is depicted in Figure 3.2. We start with a text corpus on which we apply tokenization and word sense disambiguation (WSD). Tokenization on its own results in a sequence of tokens. WSD further results in a *disambiguated corpus*, an enriched form of the sequence of tokens, whereby there are additional sequences aligned to the initial sequence of tokens. In this work we use the following additional sequences: lemmas, concepts and grammar types. The grammar type assigns a part-of-speech identifier to each token (e.g. article, adjective, adverb, noun, proper noun, punctuation mark); we use these for filtering, but not for generating embeddings. Since some tokens may have no associated lemma or concept, we pad these sequences with $\varnothing_L$ and $\varnothing_C$, which are never included in the vocabulary V.

The disambiguated corpus can optionally be modified or **filtered** in different ways. In our evaluation, we experiment with a filter whereby we (i) remove elements from the sequences if they have grammar type `article`, `punctuation mark` or `auxiliary verbs` and (ii) generalize tokens with grammar type `entity` or `person proper noun`, which replaces the original token with special tokens `grammar#ENT` and `grammar#NPH` respectively. The intuition behind this filter is that it produces sequences where each element is more semantically meaningful, since articles, punctuation marks and auxiliary verbs are binding words which should not contribute much meaning to their co-occurring words. Similarly, in many cases, we are not interested in deriving embeddings for entities (names of people, places or organizations); furthermore, many entity names may only occur a few times in a corpus and may refer to different real world individuals.

To generate embeddings for both semantic and lexical entries, we iterate through the disambiguated corpus to decide on a vocabulary V and calculate a representation of D called a co-occurrence matrix M, which is a $|V| \times |V|$ matrix, where each element $x_{ij} = \#(x_i, x_j)$. We follow word2vec, GloVe and Swivel in using a *dynamic context window* [28], whereby co-occurrence counts are weighted according to the distance between the center and the context entry using the harmonic function. More formally, we use

$$\#_\delta(x_i, x_j) = \sum_{c \in p(x_i, x_j)} \frac{W - \delta_c(x_i, x_j) + 1}{W} \qquad (1)$$

where $\delta_c(x_i, x_j)$ is the distance, in token positions, between the center entry x_i and the context entry x_j in a particular context at position c in the corpus. W is the window size as presented in section 3.1.

In standard word embedding algorithms, there is only one sequence of tokens; hence $1 <= \delta_c(x_i, x_j) <= W$. In our case we have three aligned sequences: tokens, lemmas and concepts. Hence $\delta(x_i, x_j)$ may also be 0, e.g. when x_i is a lemma and x_j is its disambiguated concept. Hence, in this work, we use a slightly modified version:

$$\delta'_c(x_i, x_j) = \begin{cases} \delta_c(x_i, x_j) & \text{if } \delta_c(x_i, x_j) > 0 \\ 1 & \text{if } \delta_c(x_i, x_j) = 0 \end{cases} \qquad (2)$$

This gives us $\#_{\delta'}(x_i x_j)$ and, based on the co-occurrence matrix M we thus apply the training phase of a slightly modified version of the Swivel algorithm to learn the embeddings for the vocabulary. The original Swivel loss function is given by:

[9] In principle, we could define two vocabularies, one for the center entries and another for the context entries; however in this paper we assume both vocabularies are equal, hence we do not make a distinction.

[10] Sensigrafo, Expert System's knowledge graph: https://www.expertsystem.com/products/cogito-cognitive-technology/semantic-technology/knowledge-graph

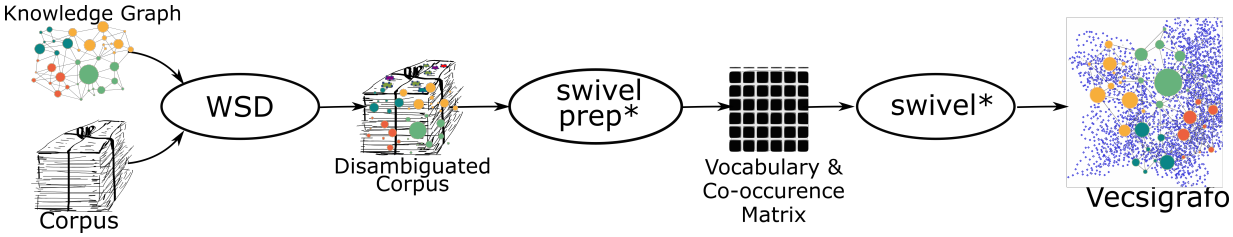

Fig. 1. Process for Vecsigrafo generation from a text corpus.

$$\mathcal{L}_S = \begin{cases} \mathcal{L}_1 & \text{if } \#(x_i, x_j) > 0 \\ \mathcal{L}_0 & \text{if } \#(x_i, x_j) = 0, \text{ where} \end{cases}$$

$$\mathcal{L}_1 = \frac{1}{2}\#(x_i, x_j)^{1/2}(\vec{x_i}^\top \vec{x_j} - \log \frac{\#(x_i, x_j)|D|}{\#(x_i)\#(x_j)})$$

$$\mathcal{L}_0 = \log[1 + \exp(\vec{x_i}^\top \vec{x_j} - \log \frac{|D|}{\#(x_i)\#(x_j)})]$$

Our modifications are: using $\#_{\delta'}(x_i, x_j)$ instead of the default definition and the addition of a vector regularization term as suggested by Duong et al. [41] (equation 3) which aims to reduce the distance between the column and row (i.e. center and context) vectors for all vocabulary elements, i.e.

$$\mathcal{L} = \mathcal{L}_S + \gamma \sum_{x \in V} \|\vec{x}_F - \vec{x}_C\|_2^2 \quad (3)$$

This modification is useful in our case, when the row and column vocabularies are the same; where in the end you would typically take the sum or the average of both vectors as your final embeddings.

3.3. Implementation

We have used Expert System's proprietary Cogito pipeline to *tokenize and disambiguate* the corpora discussed in this paper. The disambiguation algorithm is proprietary; however, internal evaluations show high accuracy on small test corpora. Expert System uses a proprietary KG called **Sensigrafo**, which is similar to WordNet, but larger and tightly coupled to the Cogito disambiguator (i.e. the disambiguator uses intricate heuristic rules based on lexical, domain and semantic rules to do its job). More specifically, we have used version 14.2 of Sensigrafo, which contains about 400K lemmas and 300K concepts (called *syncons* in Sensigrafo) interlinked via 61 relation types. Sensigrafo provides a *glossa* –a human readable textual definition– for each concept, which is only intended for facilitating the inspection and curation of the KG. In the evaluations below, we study the effect of applying alternative disambiguation algorithms[11] for comparison. We implemented three disambiguation methods: (i) the Shallow Connectivity disambiguation (`scd`) algorithm introduced by [35], which essentially chooses the candidate concepts that are better connected to other candidate concepts according to the underlying KG; (ii) the most frequent disambiguation (`mostfreqd`), which chooses the most frequent concept associated to each lemma encountered in the corpus and (iii) the random concept candidate disambiguation (`rndd`), which selects a random concept for each lemma encountered in the corpus. Note that `rndd` is not completely random, it still assigns a plausible concept to each lemma, since the choice is made out of the set of all concepts associated to the lemma.

To implement our approach, we extend the matrix construction phase of the Swivel [31] algorithm[12] to generate a co-occurrence matrix which can include both lexical and semantic entries as part of the vocabulary. Table 1 provides an example of different tokenizations and disambiguations for a context window derived from the same original text. To get a feeling for concepts and the effect of different disambiguators, notice that Cogito assigns concept #82073 (with glossa *appropriate for a condition or occasion* and synonyms *suitable*, *right*) to "proper", while `scd` and `mostfreqd` have assigned concept #91937 (with glossa *marked by suitability or rightness or appropriateness* and synonyms *kosher*). The `rndd` disambiguation has assigned an incorrect concept #189906 from mathematics (with glossa *distinguished from a weaker relation by excluding...*).

[11] See [42] for a comprehensive survey on the topic.

[12] As implemented in https://github.com/tensorflow/models/tree/master/research/swivel

Table 1

Example tokenizations for the first window of size $W = 3$ for sentence: "With regard to enforcement, proper agreements must also be concluded with the Eastern European countries...". First we show standard swivel tokenization, next we show the standard Cogito tokenization with sequences for plain text, lemmas, syncons and grammar type; next we show alternative disambiguation syncons for the same tokenization. Finally, we show cogito tokenization after applying filtering

context	t^{i-3}	t^{i-2}	t^{i-1}	t^{i}	t^{i+1}	t^{i+2}	t^{i+3}	
t_{swivel}	With	regard	to	enforcement,	proper	agreements	must	also
t	with regard to	enforcement	,	proper	agreements	must	also	be
l	with regard to	enforcement	∅	proper	agreement	must	also	be
s	en#216081	en#4652	∅	en#82073	en#191513	∅	en#192047	∅
g	PRE	NOU	PNT	ADJ	NOU	AUX	ADV	AUX
$s_{mostfreqd}$	en#216081	en#4652	∅	en#91937	en#191513	en#77903	en#191320	en#77408
s_{scd}	en#216081	en#4652	∅	en#91937	en#191513	en#239961	en#191320	en#134549
s_{rndd}	en#216081	en#4652	∅	en#189906	en#191513	en#101756782	en#191320	en#77445
t_f	with regard to	enforcement	proper	agreements	also	concluded	eastern	european
l_f	with regard to	enforcement	proper	agreement	also	conclude	eastern	European
s_f	en#216081	en#4652	en#82073	en#191513	en#192047	en#150286	en#85866	en#98025
g_f	PRE	NOU	ADJ	NOU	ADV	VERB	ADJ	ADJ

Table 1 also introduces notation we will use throughout the paper to identify embedding variations. We will use t to refer to plain text tokens and assume Cogito-based tokenization, if a different tokenization is meant, we will add a suffix like in the table to show that swivel tokenization has been used. Similarly, we use l to refer to lemmas and s to refer to concept identifiers (we assume syncons since most of our experiments use Sensigrafo, although in some cases this may refer to other concept identifiers in other KGs such as BabelNet). As described above, the source sequences may be combined, which in this paper means combinations ts or ls. Finally we use suffix _f to show that the original sequence was filtered based on grammar type information as described above.

4. Evaluation

Our approach requires a few changes to conventional algorithms for learning word embeddings, in particular the tokenization and lemmatization required to perform disambiguation affects the vocabulary. Furthermore, the introduction of concepts in the same vector space can affect the quality of word embeddings. Obviously the whole point of such a hybrid approach is to be able to learn both high quality word and concept embeddings. Hence, we posit the following research questions:

- **RQ1** What is the optimal configuration for generating Vecsigrafo embeddings for a specific corpus size? In particular, which tokenization, disambiguation and filtering combination is likely to result in better embeddings?
- **RQ2** How does Vecsigrafo compare to conventional word embeddings? More specifically:
  - **RQ2.1** Does inclusion of concepts in the same space affect the quality of the word embeddings?
  - **RQ2.2** We know that corpus size affects the quality of word embeddings, does this effect change for Vecsigrafo-based embeddings?
  - **RQ2.3** How does Vecsigrafo (based on Swivel) compare to other word-embedding algorithms
- **RQ3** How do corpus-based derived embeddings compare to other concept-embeddings such as KG-derived embeddings and lexical specificity-derived embeddings?

In an attempt to find answers to the research questions, we performed two studies on the resulting embeddings: an **ablation study** to determine the effect of different choices in our algorithm , as well as of key components like the underlying KG and disambiguator (i.e. RQ1 and RQ2.2), and a **comparative study** to compare Vecsigrafo to other word-embedding algorithms (i.e. RQ2 and RQ3). In both studies, we rely on several tasks that provide an indication about the quality of the embeddings:

- **Word similarity:** We analyze results for 14 word-similarity datasets for word and concept

relatedness. Besides testing on the embedding agreement with human-labeled gold standards, we also check inter-agreement between embeddings generated via different methods, which is a good indicator that the resulting embeddings are converging. Inter-agreement also provides evidence about how much the resulting word embeddings learned in Vecsigrafo differ from conventional word embeddings.

- **Word prediction:** We use a test corpus to model how well the resulting embeddings predict a word based on its context words, essentially recreating the word2vec loss function on a test corpus unseen during training. This task provides insight into the quality of the resulting embeddings. Also, since this task provides information about a subset of the vocabulary, it can be used to generate plots which provide an overview of possible disparities in the quality of common and uncommon words.
- **Relation prediction:** While word-similarity and word-prediction tasks are intrinsic evaluations, ultimately the goal of learning hybrid concept embeddings is to be able to refine knowledge representations. One such refinement is the prediction of specific relations in a knowledge graph.

4.1. Sources: corpora and knowledge graphs

For better comparability, we have tried to generate embeddings using available code and the same input corpus whenever possible, although in some cases we have relied on pretrained word embeddings. In this section, we first describe the corpora we have used as well as those third parties have reported using for generating pretrained embeddings. Then, we also describe how we have generated embeddings, including relevant metadata and training parameters.

Table 2 provides an overview of the corpora used for generating embeddings. To study the effect of the corpus size (and domain of the input corpus), we have used the United Nations corpus [43] as an example of a medium sized corpus that is domain specific. This corpus consists of transcriptions of sessions at the United Nations, hence the contents are domain specific with topics in politics, economics and diplomacy being predominant. We have used the English part of the corpus that is aligned with Spanish[13]. As an example of a larger corpus, we have used the dump of the English Wikipedia from January 2,018. Embeddings provided by third parties include the UMBC corpus [45], a web-corpus of roughly the same dimensions as the Wikipedia corpus. To compare our embeddings to those trained on a very large corpus, we use pretrained GloVe embeddings that were trained on CommonCrawl[14].

Table 2
Evaluation corpora

Corpus	tokens	unique	freq
europarlv7 en-es en	51.8M		
UNv1.0 en-es en	517M	2.7M	469K
wiki-en-20180120	2.89B	49M	5M
UMBC	2.95B		
CommonCrawl	840B		

Besides the text corpora, the tested embeddings contain references to concepts defined in two KGs. The first KG is a vanilla version of Sensigrafo, our proprietary semantic network, released with Cogito Studio 14.2[15]), which contains around 400K lemmas and 300K concepts (syncons). Sensigrafo is similar to WordNet, being the result of person-decades of continuous curation by a team of linguists. Like WordNet, the core relation between concepts is that of hypernymy, and both include various other lexical and semantic relations (categorical, meronymy, synonymy and conceptual similarity), although these are organized differently. One difference is that Sensigrafo contains a larger number of relation types (61 instead of 27 for WordNet 3.0) and contains several positional and prepositional relations which do not have an equivalent in WordNet. Another difference with WordNet is that Sensigrafo has explicit identifiers for concepts, while WordNet has no such identifiers; instead, WordNet uses a set of synonyms (each of which is a word sense) which refer to the same concept. The second semantic network we use in our experiments is BabelNet 3.0, which has about 14 million concepts (7 million of which are named entities). We have not trained embeddings on top of BabelNet, although we have included BabelNet derived embeddings [24, 35] in our studies.

[13] Cross-lingual applications of the embeddings is not in the scope of this paper, although we discuss some initial applications in [44].

[14] http://commoncrawl.org

[15] http://www.expertsystem.com/products/cogito-cognitive-technology

4.2. Instruments: word similarity tasks and datasets

Word similarity tasks using human judgment, as originally introduced by Resnik [46] for structured knowledge, are one of the most common intrinsic evaluations that are used to evaluate the quality of embeddings [27]. Although there are issues with these types of tasks [47, 48], they tend to provide insights into how well learned embeddings capture the perceived semantic relatedness between words. One of our hypotheses is that introducing concepts to the vector space should help to learn embeddings that better capture word similarities; hence this type of evaluation should prove useful. Furthermore, it is possible to extend the default word-similarity task –whereby the cosine similarity between the vectors of a pair of words is compared to a human-rated similarity measure– by calculating a *concept-based word similarity measure*: in this case, we select the maximum similarity between the concepts associated to the initial pair of words. This intuitively makes sense since, presumably, when a pair of words is related, human raters naturally disambiguate the senses that are closest rather than taking into account all the possible senses of the words. As a matter of notation, we will append a suffix _c to the embedding identifier when we present word similarity results that have been derived using this concept-based similarity measure.

When using word similarity datasets, there are a few details that are often not explicitly mentioned, but that can have a big effect on the results. First, there is the issue of *pre-processing* word pairs: some evaluation implementations will normalize the words, for example by lower-casing all words; in our studies we do not apply such normalizations, since different words in the vocabulary may match such normalized words and it is not clear which vector should then be used. A second related detail pertains handling missing values. Some implementations (especially vectorized implementations) will assign a default vector to words in the dataset which are not included in the vocabulary. This obviously will tend to degrade the results of embeddings where the vocabulary is small since they are no longer comparing the intended word pairs. On the other hand, it is more accurate to only include word pairs where both words are also in the vocabulary; however this means we also have to report how many of the word pairs have been used to produce a result, in order to be able to compare results with each other. For an example of how much this can impact results, see Figure 2, where the curves are vectorised results which use a default embedding for missing words and the horizontal lines are the non-vectorised results which ignore pairs with missing words. The figure shows that as the coverage percentage drops, the results suffer greatly when using the vectorised evaluation method.

We next describe the 14 word similarity datasets that we are using in our evaluations as well as a syncon similarity dataset we generated. Results will be presented in the study sections below.

4.2.1. Word-similarity datasets

The RG-65 dataset [49] was the first one generated in order to test the distributional hypothesis. Although it only has 65 pairs, the human ratings are the average of 51 raters. MC-30 [50] is a subset of RG-65, which we include in our studies in order to facilitate comparison with other embedding methods. The pairs are mostly nouns.

Another classic word similarity dataset is WS-353-ALL [51] which contains 353 word pairs. 153 of these were rated by 13 human raters and the remaining 200 by 16 subjects. The pairs are mostly nouns, but also include some proper names (people, organizations, days of the week). Since the dataset mixes similarity and relatedness, [52] used WordNet to split the dataset into a WS-353-REL and WS-353-SIM containing 252 and 203 word pairs respectively (some unrelated word pairs are included in both subsets).

YP-130 [53] was the first dataset focusing on pairs of verbs. The 130 pairs were rated by 6 human subjects. Another dataset for verbs is VERB-143 [54] which contains verbs in different conjugated forms (gerunds, third person singular, etc.) rated by 10 human subjects. The most comprehensive dataset for verbs is SIMVERB3500 [55] consisting of 3,500 pairs of verbs (all of which are lemmatized), which were rated via crowdsourcing by 843 raters and each pair was rated by at least 10 subjects (over 65K individual ratings).

MTurk-287 [56] is another crowdsourced dataset focusing on word and entity relatedness. The 287 word pairs include plurals and proper nouns and each pair was rated on average by 23 workers. MTurk-771 [57] also focuses on word relatedness and was crowdsourced with an average of 20 ratings per word pair. It contains pairs of nouns and rare words were not included in this dataset.

MEN-TR-3K [58] is another crowd-sourced dataset which combines word similarity and relatedness. As opposed to previous datasets, where raters gave an explicit score for pair similarity, in this case raters had to

make comparative judgments between two pairs. Each pair was rated against 50 other pairs by the workers. The dataset contains mostly nouns (about 81%), but also includes adjectives (about 13%) and verbs (about 7%), where a single pair can mix nouns and adjectives or verbs. The selected words do not include rare words.

SIMLEX-999 [59] is a crowd-sourced dataset that explicitly focuses on word similarity and contains (non-mixed) pairs of nouns (666), adjectives (111) and verbs (222). This dataset also provides a score of the level of abstractness of the words. Since raters were explicitly asked about similarity and not relatedness, pairs of related –but not similar– words, receive a low score. The 500 raters each rated 119 pairs and each pair was rated by around 50 subjects.

RW-STANFORD [60] is a dataset that focuses on rare (infrequent) words. Words still appear in WordNet (to ensure they are English words as opposed to foreign words). Each of the 2,034 pairs was rated by 10 crowd-sourced workers. The dataset contains a mix of nouns (many of which are plurals), verbs (including conjugated forms) and adjectives.

Finally, SEMEVAL17 (English part of task 2) [61] provides 500 word pairs, selected to include named entities, multi-words and to cover different domains. They were rated in such a way that different types of relations (synonymy, similarity, relatedness, topical association and unrelatedness) align to the scoring scale. The gold-standard similarity score was provided by three annotators.

4.2.2. Syncon-similarity dataset

Using concept-based evaluation, we can re-use the similarity rankings provided by word-similarity datasets to evaluate the quality of concepts. However, it would be better to have a dataset specifically of syncon pairs that have been ranked for similarity. Unfortunately, such a dataset would be KG-specific and we do not have a human rated dataset for Sensigrafo syncons. To still be able to generate such a dataset, we have chosen to use embeddings learned using HolE and Sensigrafo 14.2 (see section 4.4.1 for details about how the HolE embeddings were generated). The rationale is that such embeddings reflect information encoded in the KG; i.e. vectors for syncons which are connected to each other will be close in the embedding space. We initially tried sampling random pairs; however, we noticed that most pairs had a cosine similarity of around 0. This is because most syncons pairs are not directly related via KG relations, hence HolE does not assign them a specific similarity score. In order to have a meaningful dataset, instead we devised a way to generate pairs which are somehow related to each other. The resulting dataset is called `syn-sim-802` and consists of 802 pairs of syncons which HolE considers to be closely related. The 802 pairs were selected as follows:

- 300 pairs were randomly generated from the `wiki_s` vocab, as long as HolE gave them a cosine similarity score of at least 0.3. This should include various relationship types (not necessarily hypernymy).
- 200 pairs were generated by randomly picking one syncon and searching its neighbourhood for syncons with a HolE score of at least 0.4. Again, this should include various relationship types.
- 100 pairs were generated as in the previous method, but now for a HolE score of at least 0.5.
- 25 pairs were chosen from known hypernym pairs between verb concepts, as long as HolE rates them higher than 0.6.
- 25 pairs were chosen from known hypernym pairs between nouns concepts, as long as HolE rates them higher than 0.7.
- 25 pairs were chosen from known hypernym pairs between nouns, with HolE cosine similarity higher than 0.6.
- 140 pairs were hypernym pairs between nouns, with HolE cosine similarity scores between 0.3 and 0.4

The resulting dataset provides pairs of syncons which are similar to each other. The notion of similarity that is captured may also include other relationships in addition to hypernymy. 13 of the 815 pairs generated were duplicates; hence the final pair count.

4.3. Ablation study

In this part of the evaluation, we study how variations to Vecsigrafo-based embeddings affect the quality of the lexical and semantic embeddings. We do this by analyzing the various results using word-similarity tasks, which we extend to also take into account concept-similarity as discussed above.

By evaluating the word similarity datasets during the training process, we saw how the results improved during the first iterations and how the results stabilized towards the end of the training, see fig 2.[16] One of the

[16] Since some embeddings have smaller vocabularies, they were trained for less iterations overall, though in all cases we checked that training was long enough for results to converge.

things we noticed by doing this is that several of the word similarity datasets are very unstable and have a relatively high standard deviation for the same embeddings at different stages in the training. The most stable datasets were SIMVERB, SIMLEX, MEN, MTurk-771, RW, SEMEVAL and WS-353-ALL, with standard deviations ranging from .007 to .017. In the analysis below, we take into account this standard deviation σ to predict the statistical significance of any measured difference, focusing on those cases where this difference is statistically significant; i.e. we only report on differences which are bigger than 2.58σ, since this corresponds to a confidence values of 99%. [17]

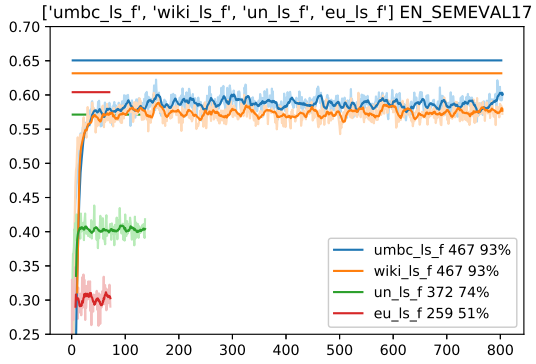

Fig. 2. Word-similarity evaluation results. The x-axis are the thousands of training steps, the y-axis are the spearman's ρ results. The lighter color curves are the actual values measured during training, while the darker color curves are smoothed values. The curves are the metrics gathered during training using a vectorized implementation. The horizontal lines are the non-vectorized results derived from the final embeddings and which do not take into account missing words from the vocabulary.

4.3.1. Vecsigrafo variations

For the ablation study, we used only variations of Vecsigrafo embeddings. We chose a relatively large maximum vocabulary size of 1.5M and all embeddings were generated with 160 dimensions. We generated embeddings for the following corpora: Europarl-v7 (`eu`), UNv1.0 (`un`) and Wikipedia (`wiki`), and for filtered and non-filtered variations: text (`t`), lemmas (`l`), concepts (`s`), text and concepts (`ts`) and lemmas and concepts (`ls`). In terms of vocabulary sizes, the `eu` embeddings had between 35K and 51K lexical embeddings and around $42 \pm 1K$ semantic embeddings; the `un` embeddings had between 22K and 32K lexical embeddings and around $80 \pm 2K$ semantic embeddings (with outliers for `mostfreqd` with 54K and `rndd` with 99K); finally the `wiki` embeddings had 1.5M lexical embeddings when no concepts were including and about 1.3M when they were included embeddings based on the cogito disambiguator had between 190K and 230K concepts, while alternative disambiguation methods resulted in 174K (`scd`), 141K (`mostfreqd`) and 215K (`rndd`).

Besides varying the filtering and combinations of sequences, we also used alternative disambiguation methods, described below.

[17] Additional materials, including result data for the ablation study, are available at: https://github.com/HybridNLP2018/vecsigrafo-paper

Table 3
Selection of statistically significant ablation differences showing effect of filtering and lemmatizing

corpus	dataset	case	base ρ	Δ
wiki	MEN	`t` vs `t_f`	.614	.072
	MTurk-771	`t` vs `t_f`	.489	.045
	SEMEVAL17	`t` vs `t_f`	.518	.102
eu	MEN	`t` vs `l`	.545	.116
		`t f` vs `l_f`	.438	.119
		`ts` vs `ls`	.494	.077
wiki	MEN	`t` vs `l`	.61	.06
	MTurk-771	`t` vs `l`	.49	.06
	SEMEVAL17	`t` vs `l`	.518	.081

4.3.2. Effect of filtering

For Europarl and the UN corpus, we could not measure any statistically significant differences between filtered and non-filtered embeddings. However, for Wikipedia, we measured an increased performance for `t_f` compared to `t` in MEN, MTurk-771 and SEMEVAL17 as shown in table 3. Although we also found some other statistically significant cases, these were on small or partial datasets (e.g. adjective or noun parts of SIMLEX-999); in all of these, filtering resulted in higher quality embeddings.

Based on these results, we can state that:

- filtering the initial sequence by removing or replacing tokens based on grammar type is helpful
- filtering is especially helpful for large corpora and when taking the plain text in the tokens (i.e. when not applying lemmatization).
- when tokens are lemmatized, filtering yields no (or statistically insignificant) improvement in the resulting embeddings.

Table 4
Selection of statistically significant ablation differences showing effect of joint lexico-semantic training

corpus	dataset	case	base ρ	Δ
eu	MEN	t vs ts	.45	.04
		t_f vs ts_f	.44	.07
		s_f_c vs ls_f_c	.48	.03
un	MEN	t vs ts	.50	.03
		l vs ls	.56	.04
		s_c vs ts_c	.56	-.04
		s_c vs ls_c	.56	-.03
wiki	MEN	t vs ts	.61	.09
		l vs ls	.67	.04
		s_c vs ts_c	.75	-.04
		s_c vs ls_c	.75	-.03

4.3.3. Effect of lemmatizing

In all three corpora, we see that using lemmas (l) rather than using the original text (t) results in better lexical (but not concept) embeddings, see Table 3. For Europarl, we measured statistically significant improvements in MEN and SEMEVAL17: l_f vs t_f and ls vs ts. For the UN corpus, we see similar improvements and for the Wikipedia corpus, we see again statistically significant improvements, but only for the case l vs t in MEN, MTurk-771 and SEMEVAL. This data shows evidence that:

- lemmatizing the tokens produces better lexical embeddings.
- lemmatization is especially important in small and medium corpora, but can have a positive impact even in large corpora.
- using lemmas instead of plain text for co-training concepts does not have a significant impact on the resulting concept embeddings.

4.3.4. Effect of joint lexico-semantic entry training

Co-training lexical and semantic embeddings always has a positive effect on lexical embeddings. However, co-training has a negative effect on concept embeddings for medium and large corpora, at least when measuring quality using the word-similarity datasets.

For the Europarl corpus, we measured improvements across all variants when comparing results of co-training vs. results from only lexical or semantic embeddings. For example, see the results in table 4 for the MEN dataset. We see similar improvements for other datasets such as SEMEVAL (only concept-based), SIMLEX and SIMVERB.

For the UN corpus, we start to see a divide: co-trained lexical embeddings always perform better than lexical embeddings trained without concept information, but concept-based word similarity tests tend to be worse for concept embeddings co-trained with lexical entries than for concept embeddings trained on their own. For example, see again table 4 for MEN. The table also shows that for the same dataset, we see some decreases for concept-based embeddings. We measured similar results for RW-STANFORD, SEMEVAL17 and SIMVERB3500.

Finally, in addition to an overall improvement due to the larger size of the corpus, for Wikipedia we see a similar trend as for the UN corpus. For MEN, we see lexical embedding improvements and concept-based semantic embedding declines as shown in table 4. We again see similar patterns for other datasets such as RW-STANFORD, SEMEVAL17 (mostly for concept-based results) and SIMVERB3500.

These results suggest that co-training lexical and concept embeddings (using a corpus):

- results in better lexical embeddings (when using a good quality disambiguator, although see below for a discussion about the effect of the disambiguation strategy)
- results in poor concept embeddings for medium to large corpora, when compared to concept embeddings trained on the same corpus, but without co-training. One caveat is that this negative result for concept embeddings may be due to the evaluation method, since we are evaluating the concept embeddings using a concept-based adaptation for the word-similarity datasets. See also the section below, where we use a concept similarity dataset specifically generated to better assess the quality of concept similarity as described in the KG, as opposed to some word-similarity measure that is extended to concepts.

4.3.5. Effect of disambiguation algorithm

In this subsection, we study the effect of the Cogito disambiguator by comparing the results with embeddings generated using alternative disambiguation methods.

To get a sense of how different the results are for the alternative disambiguation algorithms, we annotated a subset of 5,000 lines of Wikipedia with the Cogito disambiguator and with the alternative disambiguators (all using Sensigrafo 14.2). Taking the Cogito disambiguator annotations as a "standard", the Shallow Connectivity disambiguator with parameter delta 100 (as

Table 5
Selection of statistically significant ablation differences showing effect alternative disambiguation methods

corpus	dataset	case	base ρ	Δ
wiki	SEMEVAL17	`ls_f_c` vs `scd`	.71	-0.04
	MEN	`ls_f` vs `mfd`	.719	-.013
		`ls_f_c` vs `mfd`	.71	-.02
		`ls_f` vs `rndd`	.719	.021
		`ls_f_c` vs `rndd`	.711	.039

suggested in [35]) achieved 44.4 precision, 76.8 recall and 56.3 f1. Similarly, `mostfreqd` resulted in 47.9 precision, 100 recall and 64.8 f1. Finally, `rndd` achieved 29.1 precision, 100 recall and 45.1 f1. These results shows the alternative disambiguations are quite different.

Based on the word-similarity datasets and the `wiki` corpus, we see that `scd` has no significant effect on the lexical embeddings, compared to cogito disambiguation. `scd` has a slightly detrimental effect on concept embeddings, but we only measured a significant result on RW-STANFORD and SEMEVAL, as shown in table 5.

The most-frequent concept disambiguation (`mfd`) has a detrimental effect on both lexical and concept embeddings as shown for MEN in table 5. Surprisingly, `rndd`-based embeddings resulted in better lexical and concept embeddings as shown in the same table for the MEN dataset. For concept-based embeddings we also saw improvements for MTurk-771 and SIMVERB3500.

To have a better understanding of the quality of the concept embeddings, we also evaluated them using the `syn-sim-802` dataset. Table 6 shows how various embeddings correlate to this dataset. Recall that for this dataset the HolE embeddings trained on Sensigrafo serve as the baseline. We see that Cogito disambiguation results in higher correlation with HolE-based syncon embeddings, i.e. they correlate better to relations in Sensigrafo. All Cogito-based disambiguation embeddings manage to keep about 80% or more of the syncons in the vocabulary, providing good coverage of the concepts. *All the alternative disambiguation methods studied (`mostfreqd`, `rndd` and `scd` [35]) resulted in worse correlation with the HolE-based syncon embeddings and also worse coverage of the syncon vocabulary.*

The vectors derived from Shallow Connectivity disambiguation are not far behind in terms of correlation with HolE-based embeddings (we do not have any significance metrics, though); this is not surprising, since

Table 6
`syn-sim-802` results

emb	pearson	perc of pairs
HolE	1.000	100.000
wiki_ls_f	0.550	83.915
wiki_ls	0.539	77.307
umbc_ls_f	0.532	78.055
wiki_s_f	0.527	98.130
wiki_ts	0.524	80.673
wiki_ts_f	0.510	86.160
wiki_s	0.508	100.000
wiki_scd_ls_f	0.507	70.698
wiki_mostfreqd_ls_f	0.486	44.140
wiki_rndd_ls_f	0.402	86.658

`scd` uses graph relations to perform disambiguation, hence it makes sense that the vecsigrafo algorithm embeds this information. However, note that only about 70% of the syncon pairs in the original dataset could be found. An explanation for this could be that `scd` never chooses syncons which are less well connected to other syncons in the KG. Also, `scd` only uses direct relations in the KG, while the Cogito disambiguator also uses certain multi-hop relations such as those relating syncons with common domains.

Disambiguation by selecting the most frequent syncon for each lemma still manages to produce relatively good correlation with HolE-based embeddings. However, coverage suffers greatly, which was to be expected, since only one syncon is chosen per lemma.

Finally, disambiguation using a random syncon (`rndd`) from the list of available syncons for a given lemma produces the worst syncon embeddings, although coverage is maintained.

Overall, these results show –somewhat surprisingly– that:

- a good disambiguation method is not essential to get good quality lexical embeddings.
- using simple disambiguation methods such as `scd` or even `rndd` has a similar (positive) effect on the lexical embeddings as more intricate disambiguation methods such as Cogito.
- the effect of disambiguation methods on the quality of concept embeddings depends on the evaluation method. Better disambiguation methods result in better concept embeddings when evaluating them using a concept similarity dataset. No such effect could be seen using a word-similarity dataset adapted to assess concepts, suggesting such evaluation tools may be inappropriate.

4.3.6. Effect of corpus

It is well-known that larger corpora result in better word embeddings. When comparing Europarl, UN and Wikipedia embeddings, we see a confirmation of this pattern also for Vecsigrafo-based embeddings. However, we also see some differences in terms of lexical and semantic embeddings.

When going from a small corpus (Europarl) to a medium corpus (UN), we measured mostly improvements. However, we only saw significant improvements for lexical embeddings in a few datasets, while we saw significant improvements for semantic embeddings in more cases. In MEN, we saw improvements for both lexical embeddings (on average .024) and semantic embeddings (on average .036). In WS-353 and RW we only saw significant improvements for semantic embeddings (on average .06 and .05 respectively). In SEMEVAL17 we saw improvements for semantic embeddings (on average .02), but decline for lexical embeddings (on average -0.02). In SIMLEX, we only saw decline for lexical embeddings (on average -0.02).

When going from the UN corpus to Wikipedia, we saw improvements across the board. The contrast is especially clear in larger word similarity datasets such as MEN, where all variants were significantly better, both statistically and in terms of absolute values, with an average increase of .15. Concept embeddings improve more (on average .18) than lexical embeddings (on average .13). We see similar patterns in MTurk-771, RG-65, SEMEVAL17 and WS-353. Interestingly, for some datasets, we see a decline in lexical embedding quality. This is the case for SIMLEX-999, SIMVERB3500 and RW.

For SIMLEX, the trend is clear: results are best for Europarl, then decline as the corpus size grows. We think this is due to the fact that SIMLEX gives a low rating to pairs of words which are related but not similar. word2vec algorithms, in particular Swivel, are better at learning similarity for small corpora, but as the corpus grows, the embeddings start incorporating more (semantic) relatedness, which causes the performance for this particular dataset to decrease. We note that the results are not tainted by coverage issues, since even the Europarl `ls` embeddings have a coverage of 92% of the 999 pairs.

For RW, we think the results are due to the coverage of the pairs in the dataset. As can be expected, small corpora will not include enough examples of rare words, hence they are not included in the vocabulary. For the `ls` variants, the Europarl has 23% coverage, the UN has 40% and Wikipedia has 67%.

For SIMVERB3500, the issue is general poor performance of Vecsigrafo-based embeddings with spearman's ρ scores around or below 0.2. We think this may be due to the fact that many verbs in the dataset can also refer to nouns or adjectives, which may be adding noise to the word embedding.

We also generated `ls_f` embeddings for UMBC, which allowed us to compare two large corpora with each other. Our results show that UMBC results in better lexical and semantic embeddings than Wikipedia, suggesting that corpus size is not the only relevant factor. We measured statistically significant lexical embedding improvements only for RW-STANFORD (.420 vs .377) and SIMVERB (.209 vs .188). For concept-based word similarity we measured improvements in MEN (.742 vs .711), MTurk-771 (.586 vs .550), RW (.375 vs .341) and SIMVERB (.311 vs .291).

To sum up:

- Training on a medium corpus instead of a small corpus, improves both lexical and concept embeddings; concept embeddings improve more than lexical embeddings.
- Training on a large corpus instead of a medium corpus greatly improves both lexical and concept embeddings. Again concept embeddings show a higher improvement than lexical embeddings. A caveat here is that the larger corpus also increased the range of language and topics covered.
- Different large corpora also affect the embeddings, probably due to variance of topics and language style.

4.4. Comparative study

In this part of the evaluation, we study how Vecsigrafo-based embeddings compare to lexical and semantic embeddings produced by other algorithms.

4.4.1. Embeddings

Table 7 shows an overview of the embeddings used during the evaluations. We used five main methods to generate these. In general, we tried to use embeddings with 300 dimensions, although in some cases we had to deviate. In general, as can be seen in the Table, when the vocabulary size is small (due to corpus size and tokenization), we required a larger number of epochs to let the learning algorithm converge.

- Vecsigrafo based embeddings were first tokenized and word-disambiguated using Cogito. We

explored two basic tokenization variants. The first is lemma-concept with filtered tokens ("ls filtered"), whereby we only keep lemmas and concept ids for the corpus. Lemmatization uses the known lemmas in Sensigrafo to combine compound words as a single token. The filtering step removes various types of words: dates, numbers, punctuation marks, articles, proper names (entities), auxiliary verbs, proper nouns and pronouns which are not bound to a concept. The main idea of this filtering step is to remove tokens from the corpus which are not semantically relevant. We also trained a few embeddings without lemmatization and filtering. In such cases, we have kept the original surface form bound to the concept (including morphological variants) and we did not remove the tokens described above. For all the embeddings, we have used a minimum frequency of 5 and a window size of 5 words around the center word. We also used a harmonic weighting scheme (we experimented with linear and uniform weighting schemes but results did not differ substantially).

- Swivel[18] based embeddings using either a basic white-space tokenization of the input corpus, or a lemma-based tokenization performed by Cogito. We have used the default parameters defined by the open-source project. For the Wikipedia corpus, we had to reduce the number of dimensions to 256, since otherwise, the main Swivel algorithm would run out of GPU memory during training. We also imposed a limit of 1M for the vocabulary for the same reason.
- GloVe embeddings trained by us were derived using the master branch on its GitHub repository[19] and we used the default hyper-parameters defined therein.
- FastText embeddings trained by us were derived using the master branch on its GitHub repository[20] and we used the default hyper-parameters defined therein.
- HolE embeddings were trained by us using the code on GitHub[21] after we exported the Sensigrafo to create a training set of 2.5M triples including covering over 800K lemmas and syncons and 93 relations, including hypernymy relations, but also `hasLemma` relations between concepts and lemmas (We also tried to apply ProjE[22], but various errors and slow performance made it impossible to apply it to our Sensigrafo corpus.). We trained HolE for 500 epochs using 150 dimensions and the default hyper-parameters. The final evaluation after training reported an MRR of 0.13, a mean rank of 85,279 and Hits@10 of 19.48%.

Besides the embeddings trained by us, we also include, as part of our study, several pretrained embeddings, notably the GloVe embeddings for CommonCrawl –code `glove_840B` provided by Stanford[23]–, FastText embeddings based on a Wikipedia dump from 2,017 –code `ft_en`[24], as well as the embeddings for BabelNet concepts (NASARI[25] and SW2V) since these require direct access to BabelNet indices. In Table 7, we share the details that are reported by the embedding providers.

4.4.2. Word similarity results

Table 8 shows the Spearman correlation scores for the 14 word similarity datasets and the various embeddings generated based on the UN corpus. The last column in the table shows the average coverage of the pairs for each dataset. Since the UN corpus is medium sized and focused on specific domains, many words are not included in the learned embeddings, hence the scores are only calculated based on a subset of the pairs.

Table 9 shows the results for the embeddings trained on larger corpora and directly on the Sensigrafo. We have not included results for vectors trained with NASARI (concept-based) and SW2V on UMBC, since these perform considerably worse than the remaining embeddings (e.g. NASARI scored 0.487 on MEN-TR-3k and SW2V scored 0.209 for the same dataset, and see Table 10 for the overall average score). We have also not included word2vec on UMBC since it does not achieve the best score for any of the reported datasets; however, overall it performs a bit better than swivel

[18] https://github.com/tensorflow/models/tree/master/research/swivel

[19] https://github.com/stanfordnlp/GloVe

[20] https://github.com/facebookresearch/fastText/commit/3872afadb3a9f30de7c7792ff2ff1bda64242097

[21] https://github.com/mnick/holographic-embeddings/commit/c2db6e1554e671ab8e6acace78ec1fd91d6a4b90

[22] https://github.com/bxshi/ProjE

[23] http://nlp.stanford.edu/data/glove.840B.300d.zip

[24] https://s3-us-west-1.amazonaws.com/fasttext-vectors/wiki.en.vec

[25] http://lcl.uniroma1.it/nasari/files/NASARIembed+UMBC_w2v.zip

Table 7
Evaluated embeddings

Code	Corpus	Method	Tokenization	Epochs	Vocab	Concepts
	UN	vecsi	ls filtered	80	147K	76K
	UN	swivel	ws	8	467K	0
	UN	glove	?	15	541K	0
	UN	vecsi	ts	8	401K	83K
	UN	fastText	?	15	541K	0
	wiki	glove	?	25	2.4M	0
	wiki	swivel	ws	8	1.0M	0
	wiki	vecsi	ls filtered	10	824K	209K
`ft_en`	wiki	fastText	?	8	2.4M	0
	UMBC	w2v	?	?	1.3M	0
	wiki/UMBC	nasari	?	?	5.7M	4.4M
	Sensigrafo	HolE	n/a	500	825K	423K
	wiki'	fastText	?	?	2.5M	0
`glove_cc`	CommonCrawl	GloVe	?	?	2.2M	0

Table 8
Spearman correlations for word similarity datasets and **UN**-based embeddings. The column names refer to the method used to train the embeddings, the tokenization of the corpus (**l**emma, **s**yncon and or **t**ext and whether the tokens were **f**iltered), and whether **c**oncept-based word similarity was used instead of the usual word-based similarity

dataset	ft	glove	swivel	swivel l f	vecsi ls f	vecsi ls f c	vecsi ts	vecsi ts c	avg_{perc}
MC-30	0.602	0.431	0.531	0.572	0.527	0.405	0.481	**0.684**	82.5
MEN-TR-3k	0.535	0.383	0.509	0.603	**0.642**	0.525	0.558	0.562	82.0
MTurk-287	0.607	0.438	0.519	0.559	**0.608**	0.578	0.500	0.540	69.3
MTurk-771	0.473	0.398	0.416	0.539	**0.599**	0.497	0.520	0.520	94.6
RG-65	0.502	0.378	0.443	0.585	0.614	0.441	0.515	**0.664**	74.6
RW-STANFORD	0.492	0.263	0.356	0.444	**0.503**	0.439	0.419	0.353	49.2
SEMEVAL17	0.541	0.395	0.490	0.595	**0.635**	0.508	0.573	0.610	63.0
SIMLEX-999	0.308	0.253	0.226	0.303	**0.382**	0.349	0.288	0.369	96.1
SIMLEX-999-Adj	0.532	0.267	0.307	0.490	**0.601**	0.559	0.490	0.532	96.6
SIMLEX-999-Nou	0.286	0.272	0.258	0.337	**0.394**	0.325	0.292	0.384	94.7
SIMLEX-999-Ver	0.253	0.193	0.109	0.186	0.287	**0.288**	0.196	0.219	100.0
SIMVERB3500	0.233	0.164	0.155	0.231	0.306	**0.328**	0.197	0.318	94.4
VERB-143	**0.382**	0.226	0.116	0.162	0.085	-0.089	0.234	0.019	76.2
WS-353-ALL	0.545	0.468	0.516	0.537	**0.588**	0.404	0.502	0.532	91.9
WS-353-REL	0.469	0.434	0.465	0.478	**0.516**	0.359	0.447	0.469	93.4
WS-353-SIM	0.656	0.553	0.629	0.642	**0.699**	0.454	0.619	0.617	91.5
YP-130	0.432	0.350	0.383	0.456	**0.546**	0.514	0.402	0.521	96.7

but worse than vecsigrafo. For example, it achieves a score of 0.737 for MEN-TR-3k).

Table 10 shows the aggregate results. Since some of the word similarity datasets overlap —SIMLEX-999 and WS-353-ALL were split into its subsets, MC-30 is a subset of RG-65— and other datasets —RW-STANFORD, SEMEVAL17, VERB-143 and MTURK-287— have non-lemmatized words (plurals and conjugated verb forms) which penalize embeddings that use some form of lemmatization during tokenization, we take the average Spearman score over the remaining datasets. We discuss the lessons we can extract from these results in Section 5.

4.4.3. Inter-embedding agreement

The word similarity datasets are typically used to assess the correlation between the similarity of word pairs assigned by embeddings and a gold standard defined by human annotators. However, we can also use the word similarity datasets to assess how similar two

Table 9
Spearman correlations for word similarity datasets on large corpora (UMBC, Wikipedia and CommonCrawl)

corpus	sensi		umbc	wiki17	wiki18					cc	
dataset	HolE	HolE c	sw2v c	ft en	ft	glove	swivel	vecsi ls f	vecsi ls f c	glove	avg_{perc}
MC-30	0.655	**0.825**	0.822	0.812	0.798	0.565	0.768	0.776	0.814	0.786	100.0
MEN-TR-3k	0.410	0.641	0.731	0.764	0.760	0.607	0.717	**0.785**	0.773	**0.802**	99.9
MTurk-287	0.272	0.534	0.633	0.679	0.651	0.473	**0.687**	0.675	0.634	**0.693**	85.6
MTurk-771	0.434	0.577	0.583	0.669	0.649	0.504	0.587	**0.685**	0.578	**0.715**	99.9
RG-65	0.589	0.798	0.771	0.797	0.770	0.639	0.733	0.803	**0.836**	0.762	100.0
RW-STANFORD	0.216	0.256	0.395	0.487	**0.492**	0.124	0.393	0.463	0.399	0.462	81.9
SEMEVAL17	0.475	0.655	**0.753**	0.719	0.728	0.546	0.683	0.723	0.692	0.711	81.8
SIMLEX-999	0.310	0.380	**0.488**	0.380	0.368	0.268	0.278	0.374	0.420	0.408	99.4
SIMLEX-999-Adj	0.246	0.201	0.556	0.508	0.523	0.380	0.323	0.488	**0.564**	**0.622**	99.5
SIMLEX-999-Nou	0.403	0.484	**0.493**	0.410	0.383	0.321	0.331	0.422	0.464	0.428	100.0
SIMLEX-999-Ver	0.063	0.133	**0.416**	0.231	0.233	0.105	0.103	0.219	0.163	0.196	97.7
SIMVERB3500	0.227	0.318	**0.417**	0.258	0.288	0.131	0.182	0.271	0.331	0.283	98.8
VERB-143	0.131	-0.074	-0.084	0.397	**0.452**	0.228	0.335	0.207	0.133	0.341	75.0
WS-353-ALL	0.380	0.643	0.597	0.732	**0.743**	0.493	0.692	0.708	0.685	0.738	98.5
WS-353-REL	0.258	0.539	0.445	0.668	**0.702**	0.407	0.652	0.649	0.609	0.688	98.2
WS-353-SIM	0.504	0.726	0.748	0.782	**0.805**	0.615	0.765	0.775	0.767	0.803	99.1
YP-130	0.315	0.550	**0.736**	0.533	0.562	0.334	0.422	0.610	0.661	0.571	98.3

embedding spaces are. We do this by collecting all the similarity scores predicted for all the pairs in the various datasets and calculating the Spearman's ρ metric between the various embedding spaces. We present the results in Figure 3; darker colors represent higher inter-agreement between embeddings. E.g. we see that `wiki17 ft` has high inter-agreement with `ft` and very low with `HolE c`. We discuss these results in sections 5.2 and 5.3.

4.4.4. Word-concept prediction

One of the disadvantages of word similarity (and relatedness) datasets is that they only provide a single metric per dataset. In [44] we introduced Word-prediction plots, a way to visualize the quality of embeddings by performing a task that is very similar to the loss objective of word2vec. Given a test corpus (ideally different from the corpus used to train the embeddings), iterate through the sequence of tokens using a context window. For each center word, take the (weighted) average of the embeddings for the context tokens and compare it to the embedding for the center word using cosine similarity. If the cosine similarity is close to 1, this essentially correctly predicts the center word based on its context. By aggregating all such cosine similarities for all tokens in the corpus we can (i) plot the average cosine similarity for each term in the vocabulary that appears in the test corpus and (ii) get an overall score for the test corpus by calculating the (weighted by token frequency) average over all the words in the vocabulary.

Table 11 provides an overview of the test corpora we have chosen to generate word and concept prediction scores and plots. The corpora are:

- webtext [62] is a topic-diverse corpus of contemporary text fragments (support fora, movie scripts, ads) from publicly accessible websites, popular as training data for NLP applications.
- NLTK gutenberg selections[26] contains a sample of public-domain literary texts by well-known authors (Shakespeare, Jane Austen, Walt Whitman, etc.) from Project Gutenberg.
- Europarl-10k. We have created a test dataset based on the Europarl [63] v7 dataset. We used the English file that has been parallelized with Spanish, removed the empty lines and kept only the first 10K lines. We expect Europarl to be relatively similar to the UN corpus since they both provide transcriptions of proceedings in similar domains.

Figure 4 shows the word prediction plots for various embeddings and the three test corpora. Table 12 shows (i) the token coverage relative to the embedding vocab-

[26] https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/packages/corpora/gutenberg.zip

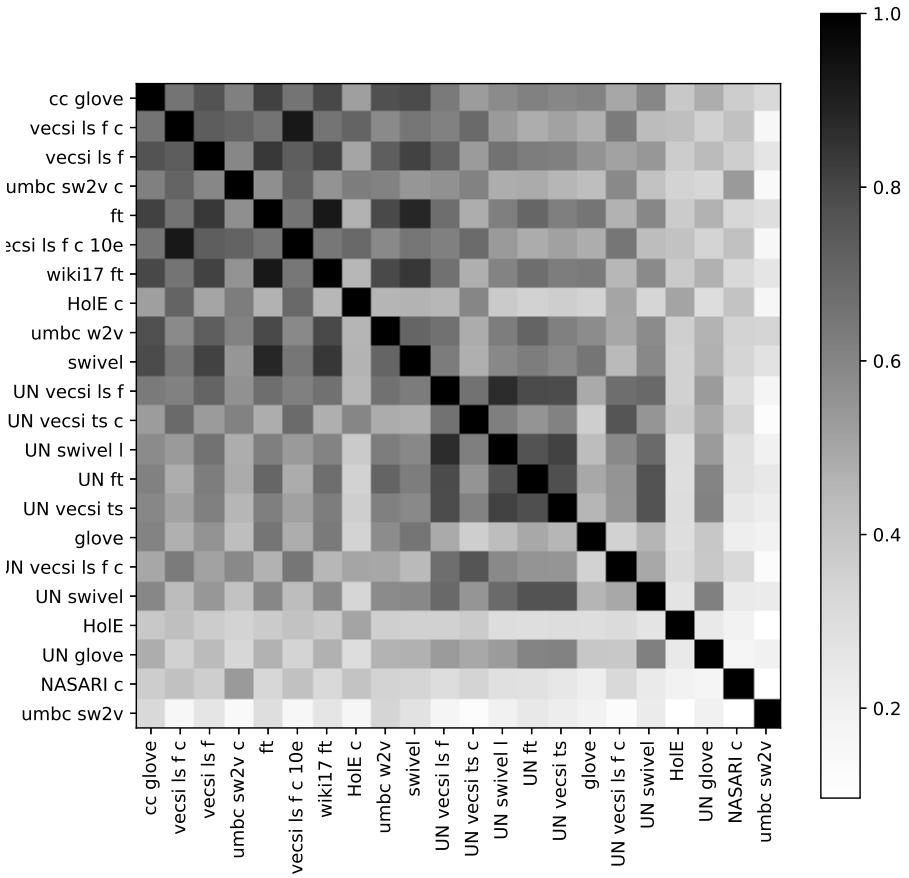

Fig. 3. Inter-embedding agreement for the word similarity datasets in the same order as Table 10. Embeddings that do not mention a corpus, were trained on Wikipedia 2,018.

ulary (i.e. the percentage of the embedding vocabulary found in the tokenized test corpus); (ii) the weighted average score, this is the average cosine similarity per prediction made (however, since frequent words are predicted more often, this may skew the overall result if infrequent words have worse predictions.); (iii) the "token average" score, this is the average of the average score per token. This gives an indication of how likely it is to predict a token (word or concept) given its context if a token is selected from the embedding vocabulary at random, i.e. without taking into account its frequency in general texts. As with previous results, we will draw conclusions about these results in section 5.

4.4.5. Relation prediction

Word (and concept) similarity and prediction tasks are good for getting a sense of the embedding quality. However, ultimately the relevant quality metric for embeddings is whether they can be used to improve the performance of systems that perform more complex tasks such as document categorization or knowledge graph completion. For this reason we include an evaluation for predicting specific types of relations in a knowledge graph between pairs of words, following recent work in the area [64–66]. At Expert System, such a system would help our team of knowledge engineers and linguists to curate the Sensigrafo.

To minimize introducing bias, rather than using Sensigrafo as our knowledge graph, we have chosen to use

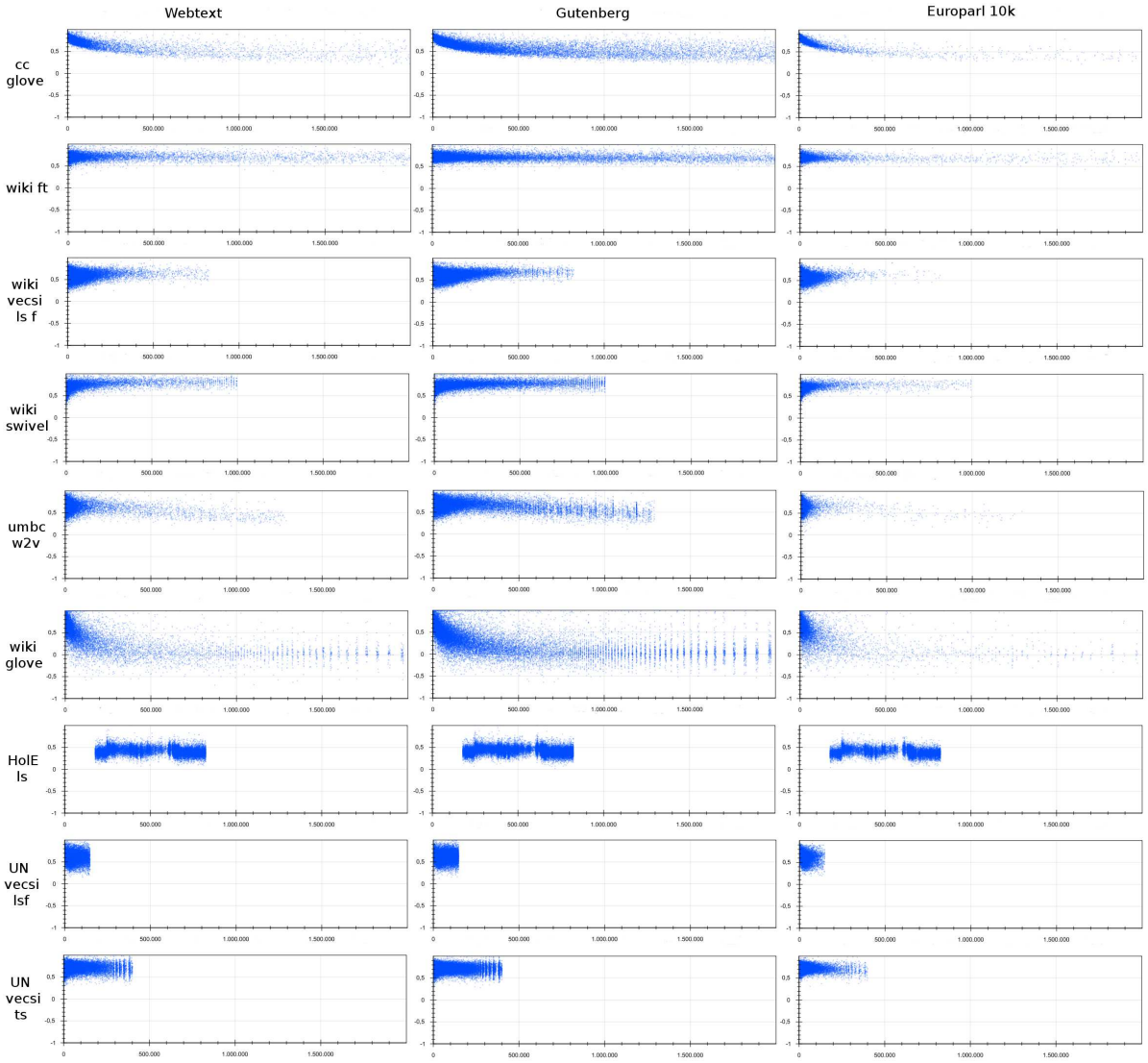

Fig. 4. Word and Concept prediction plots. The horizontal axis contains the word ids sorted by frequency on the training corpus; although different embeddings have different vocabulary sizes, we have fixed the plotted vocabulary size to 2M tokens to facilitate comparison. Since HolE is not trained on a corpus, hence the frequencies are unknown, the vocabulary is sorted alphabetically. The vertical axis contains the average cosine similarity between the weighted context vector and the center word or concept.

WordNet since we have not used it to train HolE embeddings and it is different from Sensigrafo (hence any knowledge used during disambiguation should not affect the results). For this experiment, we chose the following relations.

- `verb group`, relating similar verbs to each other, e.g "shift"-"change" and "keep"-"prevent".
- `entailment`, which describes entailment relations between verbs, e.g. "peak"-"go up" and "tally"-"count".

Datasets We built a dataset for each relation by (i) starting with the vocabulary of `UN vecsi ls f` (the smallest vocabulary for the embeddings we are studying) and look up all the synsets in WordNet for

Table 10
Aggregated word similarity results

method	corpus	avg ρ	avg coverage %
glove	cc	0.629	100.0
vecsi ls f c 25e	wiki	0.622	99.6
vecsi ls f 25e	wiki	0.619	98.6
sw2v c	umbc	0.615	99.9
ft 8e	wiki	0.613	100.0
vecsi ls f c 10e	wiki	0.609	99.6
ft	wiki17	0.606	98.9
HolE c 500e	sensi	0.566	99.6
w2v	umbc	0.566	98.9
swivel 8e	wiki	0.542	99.9
vecsi ls f 80e	UN	0.538	93.1
vecsi ts c 8e	UN	0.505	97.9
swivel l f	UN	0.480	92.9
ft 15e	UN	0.451	88.6
vecsi ts 8e	UN	0.443	91.0
glove 25e	wiki	0.438	100.0
vecsi ls f c 80e	UN	0.433	83.4
swivel 8e	UN	0.403	87.9
HolE 500e	sensi	0.381	99.6
glove 15e	UN	0.364	88.6
nasari c	umbc	0.360	94.0
sw2v	umbc	0.125	100.0

Table 11
Overview - test corpora used to gather word and concept prediction data

	tokens		
corpus	text	lemmas	concepts
webtext	300K	209K	198K
gutenberg	1.2M	868K	832K
Europarl-10k	255K	148K	143K

the lemmas. Then we (ii) searched for all the connections to other synsets using the selected relations, which gives us a list of positive examples. Finally, (iii) we generate negative pairs based on the list of positive examples for the same relation (this *negative switching* strategy has been recommended in order to avoid models simply memorizing words associated to positive pairs [28]). This resulted in a dataset of 3,039 entailment pairs (1,519 positive) and 9,889 verb group pairs (4,944 positive).

Training Next, we trained a neural net with 2 fully-connected hidden layers on each dataset, using a 90 % training, 5 validation, 5 test split. The neural nets received as their input the concatenated embeddings for the input pairs (if the input verb was a multi-word like "go up", we took the average embedding of the constituent words when using word embeddings rather than lemma embeddings). Therefore, for embeddings with 300 dimensions, the input layer had 600 nodes, while the two hidden layers had 750 and 400 nodes. The output node has 2 one-hot-encoded nodes. For the HolE embeddings, the input layer had 300 nodes and the hidden layers had 400 and 150 nodes. We used dropout (0.5) between the hidden nodes and an Adam optimizer to train the models for 12 epochs on the verb group dataset and 24 epochs on the entailment dataset. Also, to further avoid the neural net to memorize particular words, we include a random embedding perturbation factor, which we add to each input embedding; the idea is that the model should learn to categorize the input based on the difference between the pair of word embeddings. Since different embedding spaces have different values, the perturbation takes into account the minimum and maximum values of the original embeddings.

Results Table 13 shows the results of training various of the embeddings: `cc glove`, `wiki ft`[27], `HolE`, `UN vecsi ls f` and `wiki vecsi ls f`. Since constructing such datasets is not straightforward [28], we also include a set of random embeddings. The idea is that, if the dataset is well constructed, models trained with the random embeddings should have an accuracy of 0.5, since no relational information should be encoded in the random embeddings (as opposed to the trained embeddings).

The main finding was that the vecsigrafo-based embeddings learnt from the medium-sized UN corpus outperform the rest at the prediction of both target relations. Surprisingly the vecsigrafo UN embeddings also outperformed the Wikipedia-based embeddings; a possible explanation for this is the greater specificity of the UN corpus compared to Wikipedia, which spans across a large variety of topics. This could provide a stronger signal for the relation prediction task since there are less potentially ambiguous entries and the model can better leverage the relational knowledge explicitly described in the KG.

[27] For GloVe and FastText only the optimal results, based on the larger corpus (cc, wiki) , are shown.

Table 12

Aggregate word prediction values. The coverage refers to the percentage of tokens (words and concepts) in the embedding vocabulary that were found in the test corpus. The "w avg" is the average cosim weighted by token frequency and "t avg" is the average cosine similarity for all the token predictions regardless of their frequency in the corpus

test corpus	webtext			gutenberg			europarl-10k		
emb	coverage	w avg	t avg	coverage	w avg	t avg	coverage	w avg	t avg
cc glove	0.007	**0.855**	**0.742**	0.016	**0.859**	0.684	0.005	**0.868**	**0.764**
wiki swivel	0.013	0.657	**0.703**	0.027	0.664	**0.718**	0.010	0.654	0.666
UN vecsi ts	0.069	**0.688**	**0.703**	0.103	0.701	0.715	0.062	0.700	**0.717**
wiki ft	0.006	0.684	0.702	0.013	**0.702**	0.712	0.004	**0.702**	0.700
umbc w2v	0.012	0.592	0.638	0.030	0.574	0.662	0.008	0.566	0.649
UN vecsi ls f	0.138	0.630	0.617	0.214	0.652	0.628	0.128	0.681	0.636
wiki vecsi ls	0.037	0.603	0.593	0.057	0.606	0.604	0.026	0.601	0.588
HolE ls	0.035	0.414	0.416	0.056	0.424	0.424	0.026	0.400	0.398
wiki glove	0.006	0.515	0.474	0.013	0.483	0.408	0.004	0.468	0.566

Table 13

Entailment and Verb Group average prediction accuracy and standard deviation over 5 training runs

	entailment	verb group
ft_wikip	$.630 \pm .022$	$.661 \pm .021$
glove_cc	$.606 \pm .008$	$.628 \pm .013$
holE_sensi	$.603 \pm .011$	$.558 \pm .009$
vecsi_un	$.684 \pm .003$	$.708 \pm .009$
vecsi_wiki	$.608 \pm .009$	$.587 \pm .032$
rand_en	$.566 \pm .011$	$.572 \pm .003$

5. Discussion

Based on the data gathered and presented in the previous section, we now revisit our research questions and discuss the results.

5.1. Optimal configuration for learning Vecsigrafo embeddings

The ablation study presented in the previous section, produced various insights into how various options affect the quality of the resulting embeddings. Filtering the input sequences by grammar type is optional in terms of quality of the embeddings: it never hurt the measured quality of embeddings, although improvements were only clearly significant when learning plain text tokens. However, for large corpora, **filtering has a positive effect on the coverage of concepts** that are included in the vocabulary: `wiki_ts_f` has about 12K more concepts than the `ts` version and `wiki_ls_f` has about 14K more concepts than the `ls` version.

Lemmatizing produces better results than using plain text tokens; this effect is bigger for small corpora, but is still noticeable for large corpora.

Joint lexico-semantic training always improves the lexical embeddings, but results are ambiguous for the concept embeddings. The concept-based word-similarity results from table 4 suggest concept embeddings are worse when co-training, compared to learning concept embeddings directly from the `s` sequence. However, evaluation on the `syn-sim-802` dataset suggests, that at least the `ls` variants capture semantic relations from the KG (slightly) better than the `s` variants. Another consideration in this regard is the coverage of concepts in the KG that needs to be achieved: `s` achieve the best coverage of concepts and joint-training results in fewer concepts being included in the vocabulary.

Overall, the results show that the **`ls_f` tokenization offers the best compromise in terms of resulting quality of both lexical and semantic embeddings as well as coverage.**

Obviously, since Cogito and Sensigrafo are proprietary technology, they are not available to the wider community. However, the results from table 5 suggest that most of the benefits of the Cogito disambiguator can still be retained using alternative disambiguation methods such as `scd` and baselines like `rndd`.

5.1.1. On evaluating embeddings

Our results show that evaluating lexical embeddings is fairly straightforward given the large number of word-similarity datasets which are available. Our results show that **not all word similarity datasets are equally reliable**, but MEN, MTurk-771, SEMEVAL17, RW provide fairly stable results. To a

lesser extent this is also the case for SIMLEX-999, SIMVERB and WS-353, although their ranking system and ambiguity may cause problems when using special tokenizations and filtering. We found the remaining word similarity datasets less useful in our evaluations; mostly because of their limited size which results in a large standard deviation in the results.

Evaluating concept embeddings is not as straightforward as the lexical counterpart. We used concept-based evaluation using existing word-similarity datasets, but such derived datasets failed to measure differences in concept embedding quality. This may be due to breaking assumptions of word-similarity datasets. For example, SIMVERB assumes the words are verbs, but some words can refer to both verbs and nouns, which are represented in KGs by different concepts. Furthermore, our results indicate that word ambiguity is not appropriately captured in existing word similarity benchmarks, similarly to the conclusions obtained in recent studies on polysemy [67]. In order to obtain a non degraded measure, we introduced a new dataset for concept-based word-similarity evaluation. Our results with `syn-sim-802` show it is possible to build concept similarity datasets that provide a different perspective on the quality of concept embeddings. However, it would be better to assign human ratings to these concept pairs, rather than our approximation using HolE embeddings. Also, we think it may be more profitable for the community if such effort is performed based on publicly available KGs such as WordNet.

5.2. *Vecsigrafo (and sw2v) compared to conventional word embeddings*

From tables 8 and 10 we can draw the conclusion that, for the UN corpus (a medium sized corpus):

- **co-training lemmas and concepts produces better embeddings than training them using conventional word embedding methods**. In particular we see that: $\rho_{vecsi_{lsf}} > \rho_{swivel_l} \simeq \rho_{ft} \succ \rho_{vecsi_{ts}} \succ \rho_{swivel} \simeq \rho_{glove}$ Where > means that the difference is statistically significant (t-test $p < 0.01$), $\succ$ means slightly significance ($p < 0.05$) and $\simeq$ means difference is not statistically significant. As in the ablation study, we see that for the same tokenization strategy, adding concepts significantly improves the quality of the word embeddings. The comparative study furthemore shows that just lemmatizing and filtering achieves a similar quality as that of FastText (which also performs pre-processing and uses sub-word information as discussed in section 2.1).

For larger corpora such as Wikipedia and UMBC:

- **there is no statistically significant difference between FastText, Vecsigrafo[28] or SW2V[29]**. Similarly, GloVe performs at roughly the same level as these other embeddings but requires a very large corpus such as CommonCrawl to match them.
- **FastText, Vecsigrafo and SW2V significantly outperform Standard Swivel and GloVe.**
- **both lemma and concept embeddings are of high quality for Vecsigrafo based embeddings.** For SW2V-based embeddings, concept embeddings are of high quality, but the co-trained word embeddings are of poor quality. Since both methods are similar, it is not clear why this is the case.

We were surprised to see how NASARI concept embeddings (based on lexical specificity) compare poorly to the other embeddings. This was unexpected, since results in [24] were very good for similar word-similarity tests, although restricted to a few of the smaller (and thus less stable) datasets. We note that the pre-trained embeddings we used only provide embeddings for concepts which are nouns even though the method should support concepts for verbs and other grammar types. However, even for noun-based datasets we could not reproduce the results reported in [24]: for MC-30 we measured 0.68 ρ vs 0.78 reported, for SIMLEX-999-Nou we measured 0.38 instead of 0.46 and WS-353-SIM it was 0.61 instead of 0.68. An explanation for the different results may be that we do not apply any filtering by POS, as this is not specified in the concept-based word-similarity evaluation method. Instead, we find all the concepts matching the words in the word pair and return the maximum cosine similarity, regardless of whether the concepts are nouns or verbs. Also, since we do not have access to the full BabelNet, we used the REST API to download a mapping from words to BabelNet concepts. It may be the case that [24] used an internal API which performs a more thorough mapping between words and concepts, which affects the results.

In terms of inter-embedding agreement, from Figure 3 we see that, even if those concepts are derived

[28] Either concept-based or lemma-based similarity.
[29] Concept-based only.

from a different semantic net (BabelNet and Sensigrafo), **concept-based embeddings tend to have a higher agreement with other concept-based embeddings. Similarly, word and lemma based embeddings tend to be aligned with other word-based embeddings.** Since both types of embeddings achieve high scores for word-similarity (against the gold standard), **this suggests that a hybrid word-similarity evaluation approach could yield better results**.

Furthermore, we clearly see that for the medium sized corpus, all lexical embeddings tend to have a high inter-agreement with each other, but less so with lexical embeddings trained on larger corpora. For larger corpora, both lexical and concept embeddings show high inter-agreement with other similar embeddings even when trained with other corpora. For such large corpora, we see that the method used to train the embeddings (Vecsigrafo, FastText, SW2V, etc.) or the method used to predict word similarity (word-based vs concept-based) have a higher impact on the measured inter-agreement.

From the word prediction plots (Figure 4) and results (table 12, we see very different learning patterns for the various word embedding algorithms:

- GloVe tends to produce skewed predictions excelling at predicting very high-frequency words (with little variance), but as words become less frequent the average prediction accuracy drops and variance increases. This patterns is particularly clear for GloVe trained on Common Crawl. The same pattern applies for `wiki glove`, however, the plot shows that for most words (except the most frequent ones) these embeddings barely perform better than random (average cosine similarity is close to 0). This suggests that there is an issue with the default hyperparameters, or that GloVe requires a much higher number of epochs compared to other algorithms (note we initially trained most of the embeddings with 8 epochs, but due to poor performance we increased the presented GloVe embeddings for Wikipedia to 25 epochs).
- FastText produces very consistent results: prediction quality does not change depending on word frequency.
- word2vec applied to UMBC has a pattern in between that of FastText and GloVe. It shows a high variance in prediction results, especially for very high-frequency words and shows a linearly declining performance as words become less frequent.
- Swivel with standard tokenization also shows mostly consistent predictions; however very frequent words show a higher variance in prediction quality which is almost the opposite of GloVe: some high-frequency words tend to have a poor prediction score, but the average score for less frequent words tends to be higher. The same pattern applies to Vecsigrafo (based on swivel), although it is less clear for `wiki vecsi ls`. Due to the relatively small vocabulary sizes for the studied vecsigrafos trained on the UN corpus, it is hard to identify a learning pattern when normalizing the vocabulary to 2M words.

By comparing the word-prediction results between `wiki swivel` and the three Vecsigrafo-based embeddings we can see a few counter-intuitive results.

- First, on average word prediction quality *decreases* by using Vecsigrafo, which is surprising (especially since word embedding quality improves significantly based on the word-similarity results as discussed above). One possible reason for this is that the context vector for Vecsigrafo-based predictions will typically be the average of twice as many context tokens (since it will include both lemmas and concepts). However, the results for `UN vecsi ts` would suffer from the same issue, but this is not the case. In fact, `UN vecsi ts` performs as well as `wiki swivel` at this task.
- Second, both UN-based Vecsigrafo embeddings outperform the wiki-based Vecsigrafo embedding for this task. When comparing `UN vecsi ls f` and `wiki vecsi ls`, we see that due to the vocabulary size, the UN-based embeddings had to perform fewer predictions for fewer tokens; hence maybe less frequent words are introducing noise when performing word prediction. Further studies are needed in order to explain these results. For now, the results indicate that, for the word-prediction task, Vecsigrafo embeddings based on smaller corpora outperform those trained on larger corpora. This is especially relevant for tasks such as Vecsigrafo-based disambiguation, for which standard word embeddings would not be useful.

Other results from the word-prediction study are:

- Most embeddings perform better for the gutenberg test corpus than for webtext. The only exceptions are `cc glove` and `wiki glove`. This may be a result of the size of the test corpus (gutenberg is an order or magnitude larger than webtext) or the formality of the language. We assume that webtext contains more informal language, which is not represented in either Wikipedia or the UN corpus, but could be represented in CommonCrawl. Since the average differences are quite small, we would have to perform further studies to validate these new hypotheses.
- The training and test corpora matter: for most embeddings we see that the token average for Europarl is similar or worse than for webtext (and hence worse than for Gutenberg). However, this does not hold for the embeddings that were trained on the UN corpus, which we expect to have a similar language and vocabulary as Europarl. For these embeddings –`UN vecsi ts` and `Un vecsi ls f`– the Europarl predictions are better than for the Gutenberg dataset. Here again, the GloVe-based embeddings do not conform to this pattern. Since the `wiki glove` embeddings are of poor quality, this is not that surprising. For `cc glove`, it is unclear why results would be better than for both webtext and gutenberg.
- Finally and unsurprisingly, lemmatization clearly has a *compacting effect on the vocabulary size*. This effect can provide practical advantages: for example, instead of having to search for the top-k neighbours in a vocabulary of 2.5M words, we can limit our search to 600K lemmas (and avoid finding many morphological variants for the same word).

From the verb relation prediction results in Figure 13, we see that, once again, `UN vecsi ls f` outperforms other embeddings, including `wiki vecsi ls f`. The fact that the random embeddings result in an average accuracy of around 0.55 indicates that the dataset are well formed and the results are indicative of how well the trained models would perform for new pairs of words. We can see that both tasks are relatively challenging, with the models performing at most at around 70% accuracy.

5.3. Vecsigrafo compared to KG embeddings

Table 10 shows that for KG-based embeddings, the lemma embeddings (`HolE 500e`) perform poorly, while the concept-based similarity embeddings perform relatively well (`HolE c 500e`). However, the concept embeddings learned using HolE perform significantly worse than those based on the top-performing word embedding methods (FastText on wiki and GloVe on CommonCrawl) and concept-embedding methods (SW2V and Vecsigrafo). This result supports our hypothesis that **corpus-based concept-embeddings improve on graph-based embeddings since they can refine the concept representations by taking into account tacit knowledge from the training corpus**, which is not explicitly captured in a knowledge graph. In particular, and unsurprisingly, lemma embeddings derived from KGs are of much poorer quality than those derived from (disambiguated) text corpora.

The inter-embedding agreement results from Figure 3 show that HolE embeddings have a relatively low agreement with other embeddings, especially conventional word-embeddings. Concept-based HolE similarity results have a relatively high agreement with other concept-based similarities (Vecsigrafo, sw2v and NASARI).

Results from the word-prediction task are consistent with those of the word-similarity task. HolE embeddings perform poorly when applied to predicting a center word or concept from context tokens.

In Figure 4 we see that the first 175K words in the HolE vocabulary are not represented in the corpus. The reason for this is that these are quoted words or words referring to entities (hence capitalized names for places, people) which have been filtered out due to the `ls f` tokenization applied to the test corpus. Also, we see a jump in token prediction quality around word 245K which is maintained until word 670K. This corresponds to the band of concept tokens, which are encoded as `en#concept-id`. Hence words between 175K and 245K are lemmas starting from "a" to "en" and words after 670K are lemmas from "en" to "z". This again indicates that HolE is better at learning embeddings for concepts rather than lemmas (leaf nodes in the Sensigrafo KG).

6. Conclusions and future work

In this paper we presented Vecsigrafo, a novel approach to produce corpus-based, joint word-concept

embeddings from large disambiguated corpora. Vecsigrafo brings together statistical and symbolic knowledge representations in a single, unified formalism for NLP. Our results, based on the largest and most comprehensive empirical study in the area that we are aware of, show that our approach consistently outperforms word-only, graph and other hybrid embedding approaches with a medium size and large training corpora, leveling out with sub-word approaches (FastText) only in the presence of much larger corpora.

Word embeddings have shown to learn lexical and semantic relations but, staying at the level of words, they suffer from word ambiguity and brittleness when it comes to capture the different senses in a word. As a consequence, these methods usually require very large amounts of training text. Previous lemmatization and word-sense disambiguation of the training corpora enables Vecsigrafo to capture each sense much more efficiently, requiring considerably smaller corpora while producing higher quality embeddings. In the case of graph embeddings, these approaches are limited to the knowledge explicitly described in the knowledge graph, which is just a condensed interpretation of the domain according to a knowledge engineer. Vecsigrafo, on the other hand, learns from the way language is used in actual text and uses this knowledge to complement and extend the knowledge graph. Compared to previous semantic embeddings, Vecsigrafo explicitly provides embeddings for knowledge graph concepts, can be used with different knowledge graphs, and covers not only nouns but also all the lexical entries that are semantically relevant.

In this paper we have provided a detailed insight on our approach and the different aspects it comprises, including an ablation study that drills down on the effects of filtering over raw text based on grammatical information, entities and other criteria, the effects of (and different approaches to) lemmatization, the impact of jointly training lexical and semantic embeddings, the effects of applying different disambiguation strategies and the impact of training corpora of different sizes and characteristics. As the ablation study showed, looking at the improvements obtained in each specific word similarity benchmark over the different ablations confirmed that in general larger training corpus size would produce better lexical and semantic embeddings. However, the embeddings produced by large corpora like Wikipedia were outperformed by those obtained from UMBC, which is similarly sized but has a different structure, suggesting that corpus size is an important factor to consider but not the only one when it comes to balance the quality of lexico-semantic embeddings. The study also showed that jointly learning embeddings for words and concepts contributed to improve the quality of both compared to each individual case. A detailed analysis for each individual word similarity benchmark and their properties, e.g. in terms of number of contributors and similarity/relatedness pairs, contributed to identify the nuances of these conclusions, as discussed in the paper.

We also proposed two mechanisms that have proved useful to provide a deeper understanding on the quality of the resulting embeddings. Word (and concept) prediction plots allow overcoming some of the main limitation of word similarity benchmarks, which only provide a single metric per dataset, by using the embeddings to predict a word based on its context in three additional test corpora. On the other hand, inter-embedding agreement leverage the results from the word similarity benchmarks to assess how similar two embedding spaces are, allowing to identify trends over the different algorithms and corpora.

Our ongoing research seeks to enrich, validate and extend the coverage of existing knowledge graphs for NLP, as well as to explore cross-lingual, cross-modal scenarios that span across related areas of AI, with a focus on multimodal machine comprehension [68]. At Expert System we are currently applying Vecsigrafo in different tasks aimed at optimizing Cogito, assisting our team of knowledge engineers and linguists in increasingly cost-efficient ways. Some examples include the increasingly automated extension and curation of our knowledge graph, Sensigrafo, extending cross-lingual capabilities over more than 14 languages currently supported by Cogito, and enhancing the disambiguation algorithm with evidence captured from document corpora.

We hope that the algorithms and analysis reported in this paper, as well as the resources that we generated in doing so, will inspire further research to better understand and leverage the interplay between knowledge graphs and distributional semantics in cognitive tasks, supporting a new breed of hybrid methods for knowledge-based NLP. To assist this purpose, we have also created a tutorial, [30] which provides practical insight on the topics discussed herein. The tutorial follows a highly hands-on approach, with freely available sample data, teaching materials and exercises consisting of Jupyter notebooks executable online.

[30] *Tutorial on Hybrid Techniques for Knowledge-based NLP*, available at: http://expertsystemlab.com/hybridNLP18/

Acknowledgements

We gratefully acknowledge funding from projects DANTE-700367 and GRESLADIX-IDI-20160805.

References

[1] S.U.H.P. Project and E.H. Shortliffe, *MYCIN: a Knowledge-based Computer Program Applied to Infectious Diseases*, 1977.

[2] E.A. Feigenbaum, The Art of Artificial Intelligence: I. Themes and Case Studies of Knowledge Engineering, Technical Report, Stanford, CA, USA, 1977.

[3] A. Newell, The Knowledge Level, *Artif. Intell.* **18**(1) (1982), 87–127, ISSN 0004-3702. doi:10.1016/0004-3702(82)90012-1.

[4] D. Gunning, V.K. Chaudhri, P.E. Clark, K. Barker, S.-Y. Chaw, M. Greaves, B. Grosof, A. Leung, D.D. McDonald, S. Mishra and Others, Project Halo Update Progress Toward Digital Aristotle, *AI Magazine* **31**(3) (2010), 33–58.

[5] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean, Distributed Representations of Words and Phrases and their Compositionality., in: *Advances in Neural Information Processing Systems*, Vol. cs.CL, 2013, pp. 3111–3119, ISSN 10495258. ISBN 2150-8097. doi:10.1162/jmlr.2003.3.4-5.951.

[6] Y. Kim, Convolutional Neural Networks for Sentence Classification, in: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Stroudsburg, PA, USA, 2014, pp. 1746–1751. doi:10.3115/v1/D14-1181.

[7] T. Khot, A. Sabharwal and P. Clark, SCITAIL: A Textual Entailment Dataset from Science Question Answering.

[8] M.J. Seo, A. Kembhavi, A. Farhadi and H. Hajishirzi, Bidirectional Attention Flow for Machine Comprehension, *CoRR* **abs/1611.01603** (2016).

[9] A.P. Parikh, O. Täckström, D. Das and J. Uszkoreit, A Decomposable Attention Model for Natural Language Inference, in: *EMNLP*, 2016.

[10] N. Kalchbrenner and P. Blunsom, Recurrent Continuous Translation Models, in: *EMNLP*, 2013.

[11] I. Sutskever, O. Vinyals and Q.V. Le, Sequence to Sequence Learning with Neural Networks.

[12] K. Cho, B. van Merrienboer, D. Bahdanau and Y. Bengio, On the Properties of Neural Machine Translation: Encoder-Decoder Approaches, in: *SSST@EMNLP*, 2014.

[13] D. Bahdanau, K. Cho and Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, *CoRR* **abs/1409.0473** (2014).

[14] C. Olah, A. Mordvintsev and L. Schubert, Feature Visualization, *Distill* (2017), https://distill.pub/2017/feature-visualization. doi:10.23915/distill.00007.

[15] A. Sheth, S. Perera, S. Wijeratne and K. Thirunarayan, Knowledge Will Propel Machine Understanding of Content: Extrapolating from Current Examples, in: *Proceedings of the International Conference on Web Intelligence*, WI '17, ACM, New York, NY, USA, 2017, pp. 1–9. ISBN 978-1-4503-4951-2. doi:10.1145/3106426.3109448.

[16] Y. Shoham, Why Knowledge Representation Matters, *Commun. ACM* **59**(1) (2015), 47–49, ISSN 0001-0782. doi:10.1145/2803170.

[17] P. Domingos, A Few Useful Things to Know About Machine Learning, *Commun. ACM* **55**(10) (2012), 78–87, ISSN 0001-0782. doi:10.1145/2347736.2347755.

[18] M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L.S. Zettlemoyer, Deep contextualized word representations, in: *NAACL-HLT*, 2018.

[19] Y. Belinkov, N. Durrani, F. Dalvi, H. Sajjad and J.R. Glass, What do Neural Machine Translation Models Learn about Morphology?, in: *ACL*, 2017.

[20] O. Melamud, J. Goldberger and I. Dagan, context2vec: Learning Generic Context Embedding with Bidirectional LSTM, in: *CoNLL*, 2016.

[21] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, *CoRR* **abs/1810.04805** (2018).

[22] E.M. Bender, 100 Things You Always Wanted to Know about Semantics & Pragmatics But Were Afraid to Ask *, Technical Report. http://aclweb.org/anthology/P18-5001.

[23] N. Asher and A. Lascarides, Strategic conversation, *Semantics and Pragmatics* **6**(2) (2013), 1–62. doi:10.3765/sp.6.2.

[24] J. Camacho-Collados, M.T. Pilehvar and R. Navigli, NASARI: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities, *Artificial Intelligence* **240** (2016), 36–64, ISSN 00043702. ISBN 0004-3702. doi:10.1016/j.artint.2016.07.005.

[25] G.A. Miller, C. Leacock, R. Tengi and R.T. Bunker, A Semantic Concordance, *Proceedings of the Workshop on Human Language Technology - HLT '93* (1993), 303–308. ISBN 1558603247. doi:10.3115/1075671.1075742.

[26] M. Baroni and A. Lenci, Distributional Memory: A General Framework for Corpus-Based Semantics, *Computational Linguistics* **36**(4) (2010), 673–721, ISSN 0891-2017. ISBN 10.1162/coli_a_00016.

[27] M. Baroni, G. Dinu and G. Kruszewski, Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors., in: *ACL*, 2014, pp. 238–247.

[28] O. Levy, Y. Goldberg and I. Dagan, Improving Distributional Similarity with Lessons Learned from Word Embeddings, *Transactions of the Association for Computational Linguistics* **3**(0) (2015), 211–225, ISSN 2307-387X.

[29] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch and A. Joulin, Advances in Pre-Training Distributed Word Representations, *CoRR* **abs/1712.09405** (2018).

[30] J. Pennington, R. Socher and C.D. Manning, Glove: Global vectors for word representation., in: *EMNLP*, Vol. 14, 2014, pp. 1532–1543.

[31] N. Shazeer, R. Doherty, C. Evans and C. Waterson, Swivel: Improving Embeddings by Noticing What's Missing, *CoRR* **abs/1602.02215** (2016).

[32] I. Iacobacci, M.T. Pilehvar and R. Navigli, SENSEMBED: Learning Sense Embeddings for Word and Relational Similarity, in: *53rd Annual Meeting of the ACL*, 2015, pp. 95–105. ISBN 9781941643723. doi:10.3115/v1/P15-1010.

[33] X. Chen, Z. Liu and M. Sun, A Unified Model for Word Sense Representation and Disambiguation, in: *the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1025–1035. ISBN 978-1-937284-96-1.

[34] S. Rothe and H. Schütze, AutoExtend: Extending Word Embeddings to Embeddings for Synsets and Lexemes, in: *ACL*, 2015.

[35] M. Mancini, J. Camacho-Collados, I. Iacobacci and R. Navigli, Embedding Words and Senses Together via Joint Knowledge-Enhanced Training, in: *CoNLL*, 2017.

[36] A. Bordes, N. Usunier, J. Weston and O. Yakhnenko, Translating Embeddings for Modeling Multi-Relational Data, *Advances in NIPS* **26** (2013), 2787–2795, ISSN 10495258. ISBN 9780874216561. doi:10.1007/s13398-014-0173-7.2.

[37] M. Nickel, L. Rosasco and T.A. Poggio, Holographic Embeddings of Knowledge Graphs, in: *AAAI*, 2016.

[38] B. Shi and T. Weninger, ProjE: Embedding Projection for Knowledge Graph Completion, in: *AAAI*, 2017.

[39] P. Ristoski and H. Paulheim, RDF2Vec: RDF graph embeddings for data mining, in: *International Semantic Web Conference*, Vol. 9981 LNCS, 2016, pp. 498–514, ISSN 16113349. ISBN 9783319465227.

[40] M. Schlichtkrull, T.N. Kipf, P. Bloem, R. van den Berg, I. Titov and M. Welling, Modeling Relational Data with Graph Convolutional Networks, 2018.

[41] L. Duong, H. Kanayama, T. Ma, S. Bird and T. Cohn, Learning Crosslingual Word Embeddings without Bilingual Corpora, in: *EMNLP*, 2016.

[42] R. Navigli, Word Sense Disambiguation: A Survey, *ACM Comput. Surv* **41**(10) (2009). doi:10.1145/1459352.1459355.

[43] M. Ziemski, M. Junczys-Dowmunt and B. Pouliquen, The united nations parallel corpus v1. 0, in: *Language Resource and Evaluation*, 2016.

[44] R. Denaux and J.M. Gomez-Perez, Towards a Vecsigrafo: Portable Semantics in Knowledge-based Text Analytics, in: *International Workshop on Hybrid Statistical Semantic Understanding and Emerging Semantics @ISWC17*, 2017.

[45] L. Han, A.L. Kashyap, T.W. Finin, J. Mayfield and J. Weese, UMBC EBIQUITY CORE: Semantic Textual Similarity Systems, in: **SEM@NAACL-HLT*, 2013.

[46] P. Resnik, Using Information Content to Evaluate Semantic Similarity in a Taxonomy, in: *IJCAI*, 1995.

[47] M. Faruqui, Y. Tsvetkov, P. Rastogi and C. Dyer, Problems With Evaluation of Word Embeddings Using Word Similarity Tasks, in: *RepEval@ACL*, 2016.

[48] T. Schnabel, I. Labutov, D.M. Mimno and T. Joachims, Evaluation methods for unsupervised word embeddings, in: *EMNLP*, 2015.

[49] H. Rubenstein and J.B. Goodenough, Contextual correlates of synonymy, *Communications of the ACM* **8**(10) (1965), 627–633, ISSN 00010782. doi:10.1145/365628.365657.

[50] G.A. Miller and W.G. Charles, Contextual Correlates of Semantic Similarity, *Language and Cognitive Processes* **6**(1) (1991), 1–28, ISSN 14640732. ISBN 0169-0965. doi:10.1080/01690969108406936.

[51] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman and E. Ruppin, Placing search in context: the concept revisited, *ACM Transactions on Information Systems* **20**(1) (2002), 116–131, ISSN 10468188. ISBN 1581133480. doi:10.1145/503104.503110.

[52] E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Pas and A. Soroa, A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches, *Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL* (2009), 19–27, ISSN 1351-3249. ISBN 978-1-932432-41-1. doi:10.3115/1620754.1620758.

[53] D. Yang and D.M.W. Powers, Verb Similarity on the Taxonomy of WordNet, *3rd International WordNet Conference* (2006), 121–128.

[54] S. Baker, R. Reichart and A. Korhonen, An Unsupervised Model for Instance Level Subcategorization Acquisition, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)* (2014), 278–289. ISBN 9781937284961.

[55] D. Gerz, I. Vulic, F. Hill, R. Reichart and A. Korhonen, SimVerb-3500: A Large-Scale Evaluation Set of Verb Similarity, in: *EMNLP*, 2016.

[56] K. Radinsky, E. Agichtein, E. Gabrilovich and S. Markovitch, A word at a time, in: *Proceedings of the 20th international conference on World wide web - WWW '11*, ACM Press, New York, New York, USA, 2011, p. 337. ISBN 9781450306324. doi:10.1145/1963405.1963455.

[57] G. Halawi, G. Dror, E. Gabrilovich and Y. Koren, Large-scale learning of word relatedness with constraints, in: *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '12*, 2012, p. 1406, ISSN 9781450314626. ISBN 9781450314626. doi:10.1145/2339530.2339751.

[58] E. Bruni, G. Boleda, M. Baroni and N.-K. Tran, Distributional semantics in technicolor, *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics* **1**(July) (2012), 136–145, ISSN 10504729. ISBN 9781450310895. doi:10.1109/ICRA.2016.7487801.

[59] F. Hill, R. Reichart and A. Korhonen, SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation, *Computational Linguistics* **41** (2015), 665–695.

[60] M.-T. Luong, R. Socher and C.D. Manning, Better Word Representations with Recursive Neural Networks for Morphology, *CoNLL-2013* (2013), 104–113, ISSN 9781937284701. ISBN 9781937284701.

[61] J. Camacho-Collados, M.T. Pilehvar, N. Collier and R. Navigli, SemEval-2017 Task 2: Multilingual and Cross-lingual Semantic Word Similarity, in: *SemEval@ACL*, 2017.

[62] V. Liu and J.R. Curran, Web Text Corpus for Natural Language Processing, in: *EACL*, 2006.

[63] P. Koehn, Europarl : A Parallel Corpus for Statistical Machine Translation, *MT Summit* **11** (2005), 79–86, ISSN 9747431262. ISBN 9747431262. doi:10.3115/1626355.1626380.

[64] Omer Levy and Yoav Goldberg, Linguistic Regularities in Sparse and Explicit Word Representations, in: *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, Baltimore, Maryland, 2014, pp. 171–180.

[65] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury and M. Gamon, Representing Text for Joint Embedding of Text and Knowledge Bases (2015), 1499–1509.

[66] J. Weston, A. Bordes, O. Yakhnenko and N. Usunier, Connecting Language and Knowledge Bases with Embedding Models for Relation Extraction (2013), 1366–1371.

[67] H. Dubossarsky, E. Grossman and D. Weinshall, Coming to Your Senses: on Controls and Evaluation Sets in Polysemy Research, in: *EMNLP*, 2018.

[68] J.M. Gómez-Pérez, R. Denaux, A. Garcia and R. Palma, A Holistic Approach to Scientific Reasoning Based on Hybrid Knowledge Representations and Research Objects, in: *Proceedings of Workshops and Tutorials of the 9th Inter-*

national Conference on Knowledge Capture (K-CAP2017), Austin, Texas, December 4th, 2017., 2017, pp. 47–49. http://ceur-ws.org/Vol-2065/paper09.pdf.

[69] J. Bian, B. Gao and T.-Y. Liu, Knowledge-Powered Deep Learning for Word Embedding.

[70] J. Feng, M. Huang, Y. Yang and X. Zhu, GAKE: Graph Aware Knowledge Embedding, in: *COLING*, 2016.

[71] S. Lahiri, Complexity of Word Collocation Networks: A Preliminary Structural Analysis, in: *EACL*, 2014.

[72] Z. Wang, J. Zhang, J. Feng and Z. Chen, Knowledge Graph and Text Jointly Embedding, *EMNLP* **14** (2014), 1591–1601. ISBN 9781937284961.