Exploring Knowledge Graphs in an Interpretable Composite Approach for Text Entailment

Vivian S. Silva[1], André Freitas[2], Siegfried Handschuh[1,3]
[1]Department of Computer Science and Mathematics, University of Passau, Innstraße 43, 94032, Passau, Germany
[2]School of Computer Science, University of Manchester, Kilburn Building, Oxford Road, M13 9PL, UK
[3]Institute for Computer Science, University of St. Gallen, Müller-Friedberg-Strasse 8, 9000, St. Gallen, Switzerland
vivian.santossilva@uni-passau.de, andre.freitas@manchester.ac.uk, siegfried.handschuh@uni-passau.de

Abstract

Recognizing textual entailment is a key task for many semantic applications, such as Question Answering, Text Summarization, and Information Extraction, among others. Entailment scenarios can range from a simple syntactic variation to more complex semantic relationships between pieces of text, but most approaches try a one-size-fits-all solution that usually favors some scenario to the detriment of another. We propose a composite approach for recognizing text entailment which analyzes the entailment pair to decide whether it must be resolved syntactically or semantically. We also make the answer interpretable: whenever an entailment is solved semantically, we explore a knowledge base composed of structured lexical definitions to generate natural language human-like justifications, explaining the semantic relationship holding between the pieces of text. Besides outperforming well-established entailment algorithms, our composite approach gives an important step towards Explainable AI, using world knowledge to make the semantic reasoning process explicit and understandable.

Introduction

Text entailment is formally defined as a directional relationship between a pair of text expressions, denoted by T - the entailing text, and H - the entailed hypothesis. We say that T entails H if, typically, a human reading T would infer that H is most likely true (Dagan, Glickman, and Magnini 2006). It is an important input for a number of other natural language processing (NLP) tasks, such as Question Answering, Text Summarization, Information Retrieval, Machine Translation, etc., which need to deal with natural language variability to determine whether inputs and/or outputs expressed in different forms are equivalent.

Text entailment encompasses three different scenarios: (1) T and H are equivalent statements but expressed in slightly different ways; (2) H generalizes information from T; and (3) H present new information derived from T. While (1) can usually be resolved syntactically, given that only the sentence structure is altered, and (2) requires only shallow semantic information, such as synonyms and hypernyms, (3) requires knowledge that goes beyond what is expressed in T and H, demanding the use of external world knowledge to solve the entailment.

Some approaches focus on exploring the syntactic structures of T and H to determine whether they are equivalent and confirm the entailment, but can fall short of identifying more complex semantic variations. On the other hand, techniques concentrating purely on finding semantic relations between T and H will struggle to deal with pairs where only a syntactic variation holds. To overcome these issues, we propose the use of different methods to tackle different entailment scenarios, integrated as components into a composite approach that performs a *routing*, that is, it analyzes the entailment pair and sends it to the most suitable component to solve it.

For solving syntactic entailments, we use a tree edit distance algorithm over a dependency tree representation of T and H. For identifying semantic relationships, we employ a distributional (word embedding-based) navigation algorithm over graph knowledge bases composed of natural language dictionary definitions. By finding paths is these graphs linking T and H, we can provide human-readable justifications that shows explicitly what is the semantic relationship holding between them. Our contribution is twofold: besides providing a more flexible way to deal with different entailment phenomena, we also analyze different knowledge sources, showing how they compare quantitatively and qualitatively, especially from the interpretability point of view.

Related Work

Early text entailment systems relied only on word overlap and statistical lexical relations (Glickman and Dagan 2005; Pérez and Alfonseca 2005; Newman et al. 2005), while more recent approaches combine the analysis of the sentence structure with linguistic resources like WordNet, Framenet and Verbnet, which can add some shallow semantic information to the syntactic data. Edit distance (Kouylekov and Magnini 2005), alignment (Wang and Neumann 2008) and transformation (Zanoli and Colombo 2016) are some examples of such approaches. Machine learning techniques are also employed (Zhang et al. 2017), where T and H are represented as feature vectors, and multiple similarity measures (computed over lexical, syntactic and shallow semantic representations) are used to train a supervised machine learning model. Going further on the use of world knowledge, Silva et al. (2018b) proposed an approach that focuses on semantic entailments, also exploring structured knowledge bases to find semantic relationships that confirm and explain the entailment.

Recently, new techniques have been introduced for Natural Language Inference (NLI), a subtask of text entailment aimed at classifying a pair of pieces of text as *entailment*, *neutral* or *contradiction* (text entailment is originally a binary classification task, returning *yes* or *no* as an answer). With the introduction of machine learning-oriented datasets (Bowman et al. 2015; Williams, Nangia, and Bowman 2018), many inference models are now based on deep neural networks (Chen et al. 2017; Gong, Luo, and Zhang 2018), frequently using attention models (Parikh et al. 2016; Im and Cho 2017). Despite the great improvement in the quantitative results, those models are purely accuracy-driven and have increasingly more complex architectures, which leads to poor interpretability, magnifying the problem of lack of transparency and explanation already observed in most entailment systems.

Although tree edit distance has already been used in text entailment (Kouylekov and Magnini 2005), graph traversal approaches are more commonly associated with information retrieval (Frisse and others 1988; Gudivada et al. 1997), text mining (Aggarwal and Wang 2011; Runkler and Bezdek 2003), text summarization (Ganesan, Zhai, and Han 2010), and semantic similarity (Paul et al. 2016). Graphs are also used in text entailment (Kotlerman et al. 2015) as a set of potential entailment rules extracted from text, but the use of graph traversal methods for exploring independent, external resources for injecting world knowledge in the entailment recognition process is still an emerging field.

Syntactic-Semantic Composite Text Entailment

Our proposed text entailment approach is based on the notion that syntactic and semantic phenomena require different approaches to be solved. While in the first case an analysis of the structure of the sentences may suffice, in the second it is necessary to look for the semantic relationship holding between the text and hypothesis, often requiring external world knowledge. To decide what is the best approach, we employ a routing mechanism that relies on the notion of *overlap*, which will indicate whether there are terms that could be semantically related or not. We assume that a semantic relationship holds between two entities e_1 and e_2, $e_1 \neq e_2$, both referring to a third entity, which we call the *referent*. For example, consider the entailment pair 2.4 from the BPI[1] dataset:

2.4 T: North Korea launched a test missile Wednesday.
2.4 H: The missile was launched on Wednesday.

Apart from stop words, all concepts in T are also contained in H. Now consider the pair 43.1 from the same dataset:

43.1 T: A worker cleans up the streets.
43.1 H: The streets are tidy.

Here, some concepts differ from T to H. Furthermore, the entities "clean up" and "tidy" are semantically related, and both are referring to the entity "the streets".

[1] http://www.cs.utexas.edu/users/pclark/bpi-test-suite/

The overlap O is computed over the bag-of-words representation of T and H, denoted by $T' = \{t_1, t_2, ..., t_n\}$ and $H' = \{h_1, h_2, ..., h_m\}$, where t_i and h_i are tokens in T and H, respectively, and n and m are the sizes of T' and H', respectively. Therefore, $O = T' \cap H' = \{w_1, w_2, ..., w_k\}$, where k is the size of O. Three scenarios may occur:

(1) *total overlap*, where all the tokens of H' are contained in T' or (less commonly) vice-versa, that is, $k = m$ or $k = n$;
(2) *partial overlap*, where some but not all of the of tokens of T' are contained in H', so $k < n$ and $k < m$; and
(3) *null overlap*, that is, no tokens of T' are contained in H', so $O = \varnothing$ and $k = 0$

After a preprocessing stage that generates T' and H', the router computes O and decides the entailment model the pair will be sent to: if there is a partial overlap, it sends the pair to the Graph Navigation model to be solved semantically, because there are entities $e_1 \in T'$ and $e_2 \in H'$ not contained in O (so $e_1 \neq e_2$) for which a semantic relationship can hold and is likely that the referent is contained in O. For the other two scenarios, the router sends the pair to be solved syntactically by the Tree Edit Distance model, because if there is a total overlap it means that only the sentence structures of T and H are different, and if there is a null overlap, even if there are some potential candidates e_1 and e_2 that could be semantically related, it is more likely (although not certain) that they are referring to completely different entities, since there is no common referent in O.

After the entailment is solved by the suitable model, returning *yes* or *no* as the output, an interpretability module uses the evidence produced by the entailment algorithm to generate a justification aimed at explaining the algorithm's decision. The general architecture of the interpretable composite entailment approach is shown in Figure 1.

For solving entailments syntactically, we use the **Tree Edit Distance** model, which computes the minimal-cost sequence of operations (insert, delete or replace) necessary to transform the tree representation of T into the tree that represents H. We use the *All Paths Tree Edit Distance* (APTED) (Pawlik and Augsten 2016), which improves over the classical algorithm of Zhang and Shasha (1989) by being tree-shape independent. The edit distance is computed over the syntactic dependency trees of T and H, generated by the Stanford dependency parser (Chen and Manning 2014). Although what is generated by the parser is a dependency graph, it can be easily converted to an acyclic tree, where nodes with more than one incoming edge are expanded only at the first time they are referenced, and represented as childless nodes in subsequent references (similar to the string representation provided by the parser for the graph). We represent dependencies between terms, which are labeled edges in the original graph, as intermediary nodes between the two nodes they link. Figure 2 shows the graph generated by the dependency parser for the sentence "A worker cleans up the streets" and the resulting dependency tree which will be sent as input to the tree edit distance algorithm.

Since we represent dependencies as nodes in the tree, our tree edit distance model penalizes node replacement more

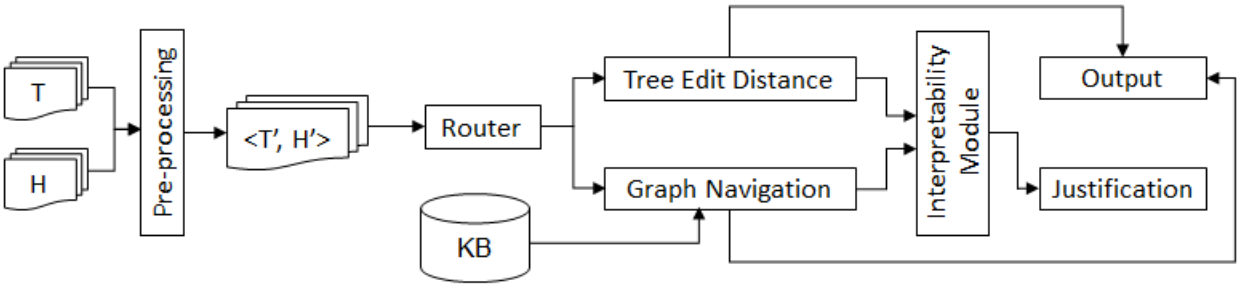

Figure 1: General architecture of the proposed interpretable composite text entailment approach

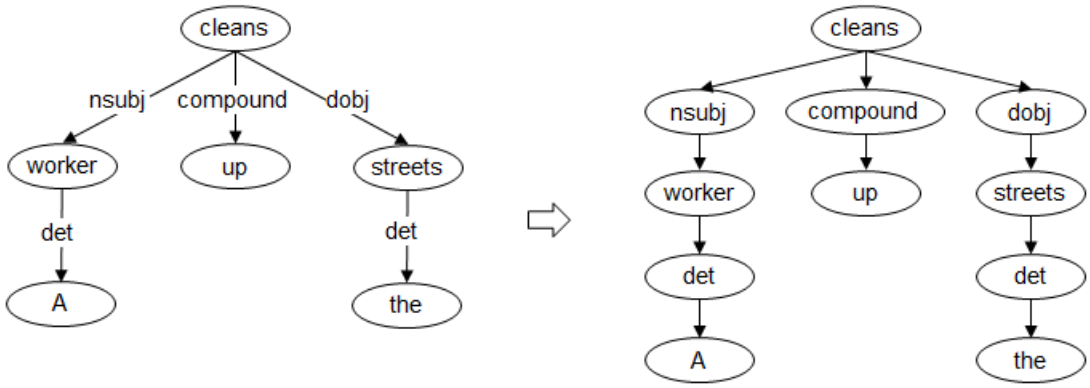

Figure 2: Dependency graph (left) and the resulting dependency tree (right) which is sent to the tree edit distance algorithm

than insertion and deletion, because replacing a node x between nodes a and b in T by a node y between the same nodes a and b in H means changing the dependency between them, or changing one of the arguments of a dependency, if the replacement comes before or after a sequence of two node a and b which are identical in T and H. This is done by a *weighted cost model* with higher weight for replacements than for insertions and deletions, and by the calculation of the relative edit distance *relDist*, which is the edit distance *dist* relative to the difference *diff* between the sizes of the two trees, given by $relDist = dist/diff$. If the two trees are roughly the same size, but many edit operations are performed, they are probably replacements, which means many dependencies and/or arguments are being changed, so *diff* is low and *relDist* increases. If approximately the same number of operations are performed for trees having different sizes (usually, T larger than H), there will be more insertions and/or deletions. In this case, *diff* is higher and *relDist* decreases, which favors scenarios where the tree for H is a subtree of the tree for T, and, therefore, insertions/deletions will occur more often and affect the validity of the entailment less than replacements. The *relDist* is then compared against a threshold t, and the pair is classified as an entailment if $relDist < t$, and as a non-entailment otherwise.

For dealing with entailments involving semantic phenomena, we use the **Graph Navigation** model introduced by Silva et al. (2018b). This model is based on a distributional navigation algorithm (DNA) (Freitas et al. 2014), which uses Distributional Semantic Models (DSMs) to perform a selective reasoning in a knowledge graph. Given a pair of terms, namely a *source* and a *target*, and a threshold η, the DNA finds all paths from source to target, with length l, formed by concepts semantically related to the target wrt η (Freitas et al. 2014). In our approach, the source is an entity $e_i \in T'$ and the target is another entity $e_j \in H'$, both e_i and e_j not contained in the overlap set O generated in the routing stage, so $e_i \neq e_j$.

As in (Silva, Freitas, and Handschuh 2018b), we use an external knowledge graph over which we perform a *depth first search algorithm*, exploring first the paths whose next node to be visited has the highest semantic similarity value wrt the target. Starting from the source node, the algorithm retrieves all its neighbors $\{x_1, x_2, ..., x_n\}$ and computes the similarity relatedness $sr(x_i, target)$. All the nodes for which $sr > \eta$ are included in the set of nodes to be visited next and each of them generates a new path. For each path, the search continues until the next node to be visited is equal to the target, or until the maximum path length is reached. The search stops if no path reaches the target before the maximum number of paths is reached. The distributional

graph navigation algorithm is schematized in Figure 3.

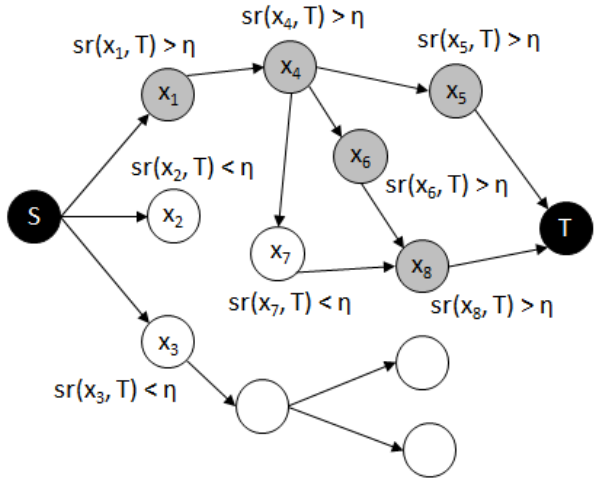

Figure 3: The distributional navigation algorithm. Gray nodes, for which $sr(x_i, T) > \eta$, make up valid paths between the source node S and the target node T. The path $\{S, x_1, x_4, x_5, T\}$ is the shortest one

Differently from Silva et al. (2018b), who uses syntactic rules to identify source-target pairs and nodes' head words, we select all the inputs for the graph navigation algorithm semantically, also using DSMs. The source-target pairs are the pairs of terms formed by the aforementioned entities $e_i \in T'$ and $e_j \in H'$, and the head words are the main words in a node of the knowledge graph that contains a multi-word expression (see next Section), and which will define the next nodes to be visited. For defining the source-target pairs, we compute the semantic similarity measures between T'' and H'', where $T'' = T' - O$ and $H'' = H' - O$, as the Cartesian product $P = T'' \times H''$. The results are then sorted and the highest scoring pairs - the set P' - are sent as the inputs for the Graph Navigation algorithm. It is done similarly for the head words, but is this case, the similarity measure is computed between each word/phrase that constitutes a node and the target node. In both cases, we remove stop words, and, to get the head words, we also remove words with low *inverse document frequency* (IDF), since they are too frequent (for example, verbs such as "get", "put", "cause" or "make") and can be reached from almost any node in the graph, leading to diverting paths. IDF is calculated using as the corpus the same linguistic resource that gave origin to the knowledge graph being explored by the algorithm (see next Section).

Word sense disambiguation comes as a natural consequence of the distributional navigation mechanism while choosing the next nodes to be visited in the graph: by looking for the word/phrases that are more semantically related to the target, the algorithm naturally selects the correct (or at least the closest) word senses, since unrelated word meanings will have lower similarity scores wrt the target, and the paths containing them will be excluded by the algorithm.

The Graph Navigation algorithm finds all paths between e_i and e_j, for each $\{e_i, e_j\} \in P'$. The shortest path is then chosen as the solution, since it provides the shortest distance between the source and the target, showing that, semantically, they are more closely related. The contents of the nodes composing this path are then used to generate the justification that explains the entailment decision. If no path is found at all, the entailment is rejected.

Exploring Lexical Knowledge for Semantic Interpretability

The ability to explain how decisions are reached is becoming a key requirement for AI systems (Gunning 2017). Although a model may produce accurate results, if it lacks transparency, not showing clearly how it is using the data, it can become harder for users to trust its predictions.

Generating natural language justifications is an important feature for increasing a system's interpretability, and the use of external sources of world knowledge can provide a both semantically rich and interpretable resource, which can support the generation of explanations. Dictionary-style definitions are a rich source of such knowledge and, different from formal structured resources like ontologies, they are domain-independent and largely available. Many NLP systems, including text entailment systems (Clark, Fellbaum, and Hobbs 2008; Herrera, Penas, and Verdejo 2006), already explore lexicons, notably WordNet (Fellbaum 1998), but they usually look only at the structured information, that is, links such as synonyms, hypernyms, etc. The natural language definition is left aside, although it contains the largest amount of relevant information about an entity: its type, essential attributes, primary functions, and often many non-essential, but very informative, attributes as well.

We rely on the knowledge provided by lexical dictionary definitions for looking for relationships between the text and the hypothesis whenever the entailment is solved semantically. These relationships not only confirm the entailment but also explain why the entailment is true. To make use of natural language definitions in our approach, we structure them, converting a whole dictionary into a knowledge graph following the methodology proposed by Silva et al. (2018a). In this model, definitions are split into *entity-centered semantic roles* (Silva, Handschuh, and Freitas 2016) and each of these roles become a node in the graph (the *role nodes*), meaning that each node encloses a self-contained amount of information regarding the entity being defined, which is also a node (the *entity nodes*). Figure 4 depicts the graph representation for one of the definitions of the concept "planet" ("any of the nine large celestial bodies in the solar system that revolve around the sun and shine by reflected light") according to this model.

When the Graph Navigation model explores this *definition knowledge graph* (DKG), it identifies the role nodes that are more semantically related to the target, get their head words (see previous Section) and use them as the next entity nodes to be visited. The resulting path is, then, composed by a sequence of entity nodes and the relevant role nodes linked to them. This path is sent to the interpretability module, which formats them into a human-readable justification,

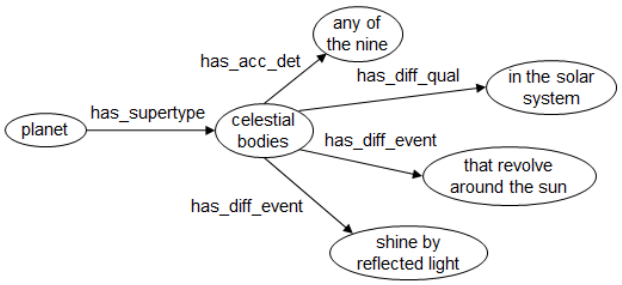

Figure 4: The graph representation of a lexical definition. The node labeled "planet" is an entity node (the entity being defined), and all the other ones are role nodes. *Supertype*, *accessory determiner*, *differentia quality* and *differentia event* are some of the semantic roles for definitions introduced in (Silva, Handschuh, and Freitas 2016)

which shows what the relationship between the source (and therefore, the text T), and the target (the hypothesis H) is, and makes clear what was the reasoning followed by the algorithm. As an example, consider the entailment pair 15.6 from the BPI dataset:

15.6 T: Baghdad has seen a spike in violence since the summer.
15.6 H: Baghdad has seen an increase in violence.
15.6 A: YES

Here, the best source-target pair is e_1 = "spike" and e_2 = "increase", while the *referent* is "violence", since both e_1 and e_2 refer to this concept. A possible path between the source and the target in a DKG is shown in Figure 5.

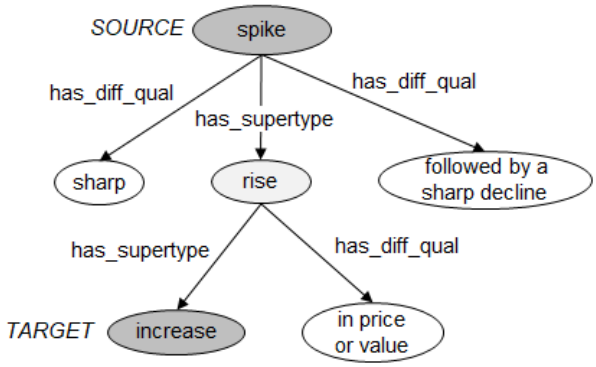

Figure 5: A path, indicated by the gray nodes, between source node "spike" and target node "increase" in a DKG

In this path, nodes are linked by the "has_supertype" property, which defines the *kind* of an entity, so the justification derived from this sequence of nodes is:

A **spike** is a kind of rise
A rise is a kind of **increase**

In order to evaluate how different resources influence the entailment results, including the justifications generated, we built three DKGs, using the definitions from WordNet, Wikitionary, and the set of definitions extracted from Wikipedia pages provided by Faralli and Navigli (2013). All definitions were filtered so potential invalid definitions could be removed (for example, verb definitions not beginning with a verb or an adverbial phrase followed by a verb, noun definitions beginning with verbs or prepositions, etc.), and, for Wikipedia, definitions for named entities were also excluded with the aid of the Stanford Named Entity Recognizer (NER), so the final content could be closer to a regular dictionary. Due to the natural limitations of the NER, many named entity definitions remained in the final set, but even so, this filter helped in setting a manageable size for the final graph, without leaving out potentially relevant information. Each set of definitions was then labeled and finally converted to an RDF graph. Table 1 shows the dimensions of the resulting graphs.

Table 1: Final dimensions of the definition knowledge graphs used in the composite text entailment approach

Resource	Noun Defs	Verb Defs	Total
WordNet	79,939	13,760	93,699
Wikitionary	390,417	73,826	464,243
Wikipedia	859,087	-	859,087

Evaluation

We evaluated our approach on two datasets: the Princeton-Boeing-ISI (BPI) and the Guardian Headlines Sample[2] (GHS). BPI has a good mix of syntactic and semantic examples, and GHS is fully composed of real-world data (headlines and sentences from The Guardian newspaper stories), without artificially assembled hypotheses. Both datasets are balanced, with half positive and half negative examples, and have 250 and 800 entailment pairs, respectively.

We compare our results with a purely syntactic entailment algorithm, the *Edit Distance* (Kouylekov and Magnini 2005) implementation provided by EOP (Magnini et al. 2014), a state-of-the-art text entailment framework; a syntactic approach that employs linguistic resources from where shallow semantic information is extracted, the *Maximum Entropy Classifier (Base+WN+TP+TPPos+TS_EN)* (Wang and Neumann 2008) using WordNet and VerbOcean, also made available by EOP; and a purely semantic approach, the Graph Navigation algorithm using WordNet definition graph, as reported by Silva et al. (2018b).

All similarity measures were computed through the Indra (Sales et al. 2018) service, using word2vector as the DSM. Table 2 shows the precision, recall and F-measure obtained by each of the Composite Entailment approach configurations, which are given by the definition graph used by the Graph Navigation component, that is, the knowledge bases derived from WordNet (WN), Wikitionary (WKT), and Wikipedia (WKP), as well as the baselines results.

[2]https://goo.gl/4iHdbX

Table 2: Evaluation results. The upper part shows the baselines, and at the bottom are the proposed composite entailment approach's results.

	BPI			GHS		
	Precision	**Recall**	**F-measure**	**Precision**	**Recall**	**F-measure**
EditDistance	0.44	0.65	0.53	0.96	0.30	0.45
MaxEntClassifier	0.46	0.57	0.51	0.50	1.00	0.66
GraphNavigation	0.65	0.54	0.59	0.56	0.50	0.53
CompositeEntailment (WN)	0.57	0.79	0.66	0.52	0.70	0.60
CompositeEntailment (WKT)	0.53	0.70	0.60	0.50	0.70	0.58
CompositeEntailment (WKP)	0.41	0.26	0.32	0.42	0.11	0.17

Among the three Composite Entailment configurations, the one that employs WordNet as the knowledge base performs better than the other ones. For the BPI dataset, the Composite Entailment present much better results than both syntactic approaches (EditDistance and MaxEntClassifier). The semantic-only GraphNavigation approach still presents better precision, but, due to the better handling of syntactic pairs and also the improved, semantic-based selection of source-target pairs and head words for multi-word expression nodes, the Composite Entailment delivers a much higher recall, leading to an overall higher F-measure.

For the GHS dataset, the EditDistance algorithm shows the highest precision but very low recall. Here again the Composite Entailment presents much higher recall and, consequently, better F-measure than both syntactic-only EditDistance and semantic-only GraphNavigation. The MaxEntClassifier shows higher F-measure, but it classifies all but two of the 800 pairs as entailment, hence the 100% recall and 50% precision, since the dataset is balanced. Given that positive pairs are structurally very different from negative ones in this dataset (in positive pairs, H is a news story short headline and T is the first, usually long, story's sentence, and in negative pairs both T and H are random, often long sentences from the same story) it is somewhat hard to grasp the MaxEntClassifier's rationale behind those decisions.

Comparing Lexical Knowledge Bases

More important than the quantitative improvements, the interpretable characteristic of the Composite Entailment approach represents a concrete gain for the final user, providing them with human-like explanations for the entailment decisions whenever a more complex semantic relationship is involved. The justifications, though, depends heavily on the knowledge base (KB) employed in the entailment recognition. To evaluate the impact of the type and quality of the KB contents in our approach, we chose three resources with different characteristics: WordNet is a lexicon built by lexicographers under a relatively controlled environment, Wikitionary is a dictionary built collaboratively by lay users, and Wikipedia is also built collaboratively but more focused on encyclopedic than lexical knowledge.

The best measure for comparing the performance of the KBs is the *recall*: the more useful information the graph contains, the more paths (meaning semantic relationships between source and target words) can be found and, consequently, more entailments can be recognized. As can be seen in Table 2, WordNet graph offers the best quantitative results, while Wikipedia shows notably bad outcomes. Despite the negligible results, it is worth listing Wikipedia numbers to highlight that, for the entailment task, the content type is more relevant than the amount of information in the graph. WordNet graph is roughly only 10% of the Wikipedia graph, but the first contains far more common nouns denoting basic language concepts, while the second, besides not defining verbs, privileges the definitions of people, places, arts and entertainment artifacts (films, books, songs, etc.) and other entities expressed by proper nouns. In fact, Wikipedia lacks definitions for many concepts present in the datasets for which relationships are sought: "violence" and "damage", "signatory" and "agreement", "decontamination" and "contaminants", or "bet" and "gamble", to name a few.

As for the Wikitionary graph, different from the Wikipedia one, which fails to provide the information to start most of the paths, most of the relevant concepts are there as much as for the WordNet graph, but the issue here refers more to the definition's completeness: if not enough information is contained in the definition, that is, the definition of an entity does not mention other entities it is essentially related to, paths will start but won't reach the target. Being built by experts, WordNet definitions tend to follow some patterns and are more prone to cover essential attributes. On the other hand, in a collaborative environment, despite the larger volume of information that can be generated, high-quality standards can't always be ensured. This is the reason why, in spite of its much larger dimensions, Wikitionary graph yields lower recall for the BPI dataset than the WordNet one. Again, the coverage and regularity of the KB contents prove to be more important than its size. An example from the BPI dataset that can be solved and justified through the WordNet graph, but not when Wikitionary or Wikipedia ones are used:

11.2 T: The UN Security Council demands that North Korea stop making nuclear weapons.
11.2 H: The Security Council wants North Korea to stop making nuclear weapons.
11.2 A: YES

Justification:
To **demand** is a way of to ask
To ask is a way of to require
To require is synonym of to **want**

From the GHS dataset:

16089 T: Frank Quattrone, a star investment banker of the dotcom era, was sentenced yesterday to 18 months in prison for obstruction of justice.
16089 H: Quattrone jailed for obstruction
16089 A: YES

Justification:
To **sentence** is to pronounce a sentence on in a court of law
A sentence is a period a prisoner is imprisoned
To imprison is synonym of to **jail**

As mentioned before, Wikipedia does not include definitions for verbs, so "to demand" and "to sentence" (the source nodes in both examples) can't be found as nodes in its graph. Wikitionary does have definitions for both "to demand" and "to sentence", but they fail to establish a (direct or indirect) link with "to want" in the first case and with "to jail" in the second example, so, while traversing its graph, no paths starting at those nodes will reach the target.

Regarding the quality of the justifications generated by each graph, we observe that WordNet and Wikipedia explanations are usually more informative than Wikitionary ones. To illustrate this pattern, consider the entailment pair 57.1 from the BPI dataset:

57.1 T: Many soldiers were killed in the ambush.
57.1 H: The soldiers were attacked by surprise.
57.1 A: YES

The justifications given by each of the graphs are as follows:

By WordNet graph: "An **ambush** is an act of concealing yourself and lying in wait to **attack** by surprise".
By Wikitionary graph: "An **ambush** is a kind of **attack**".
By Wikipedia graph: "An **ambush** is a military tactic to **attack** an unsuspecting enemy from concealed positions such as among dense underbrush or behind hilltops".

Considering the goal of finding the relation between "ambush" and "attack", all the graphs succeed, but WordNet seems to find the best balance between being not too short nor overly detailed.

In summary, Wikipedia graph provides good quality justifications, if too detailed (which is understandable given its encyclopedic nature) but yields very low recall for lacking many basic concepts. Wikitionary graph has a good coverage of relevant concepts, but recall and justification quality may be impaired by incompleteness resulting from the amateur nature of the definitions creation process. WordNet graph, despite being the smallest KB, provides the highest recall and often the most concise justifications, thanks to the focus on the description of basic language concepts and a more professional creation process which leads to better quality definitions.

Conclusion

We presented an interpretable composite approach for recognizing textual entailments that employs a routing mechanism to decide whether entailment pairs should be dealt with syntactically or semantically. For pairs predominantly showing structural differences, we use a Relative Tree Edit Distance model over dependency parse trees, and for pairs containing semantic relationships we employ a Distributional Graph Navigation model over knowledge bases composed of structured dictionary definitions. The paths found in those graph KBs are used to render the entailment interpretable, providing natural language, human-like justifications for the entailment decision.

We generated knowledge graphs from WordNet, Wikitionary and Wikipedia definition sets to assess how each of them could leverage our approach's interpretability. Given that text entailment deals with language variability, we concluded that knowledge graphs covering the most basic, everyday language concepts yield the best results, so regular dictionaries are more useful than encyclopedic KBs for this task. We also found that definitions created by lexicographers under a controlled environment tend to be more complete and, consequently, generate higher quality justifications than those created in collaborative environments by lay users. All in all, our analysis shows that the use of external world knowledge bases is a valuable feature for increasing intelligent systems' interpretability, and recent accuracy-driven advancements, such as new NLI techniques, could benefit from these resources to also stay in line with Explainable AI requirements.

Acknowledgments

This work was supported in part by a grant from the European Union Horizons 2020 – the Framework Programme for Research and Innovation (2014–2020) under grant agreement 740723 Project CSAWARE. Vivian S. Silva is a CNPq Fellow – Brazil.

References

Aggarwal, C. C., and Wang, H. 2011. Text mining in social networks. In *Social network data analytics*. Springer. 353–378.

Bowman, S. R.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics (ACL).

Chen, D., and Manning, C. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 740–750.

Chen, Q.; Zhu, X.; Ling, Z.-H.; Wei, S.; Jiang, H.; and Inkpen, D. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 1657–1668.

Clark, P.; Fellbaum, C.; and Hobbs, J. 2008. Using and extending WordNet to support question-answering. In *Proceedings of the 4th Global WordNet Conference (GWC'08)*.

Dagan, I.; Glickman, O.; and Magnini, B. 2006. The PASCAL recognising textual entailment challenge. In *Machine learning challenges: evaluating predictive uncertainty, vi-*

sual object classification, and recognising textual entailment. Springer. 177–190.

Faralli, S., and Navigli, R. 2013. A java framework for multilingual definition and hypernym extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 103–108.

Fellbaum, C. 1998. *WordNet*. Wiley Online Library.

Freitas, A.; da Silva, J. C. P.; Curry, E.; and Buitelaar, P. 2014. A distributional semantics approach for selective reasoning on commonsense graph knowledge bases. In *International Conference on Applications of Natural Language to Data Bases/Information Systems*, 21–32. Springer.

Frisse, M., et al. 1988. Searching for information in a hypertext medical handbook. *Communications of the ACM* 31(7):880–886.

Ganesan, K.; Zhai, C.; and Han, J. 2010. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd international conference on computational linguistics*, 340–348. Association for Computational Linguistics.

Glickman, O., and Dagan, I. 2005. A probabilistic setting and lexical cooccurrence model for textual entailment. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, 43–48. Association for Computational Linguistics.

Gong, Y.; Luo, H.; and Zhang, J. 2018. Natural language inference over interaction space. In *Proceedings of the Sixth International Conference on Learning Representations*.

Gudivada, V. N.; Raghavan, V. V.; Grosky, W. I.; and Kasanagottu, R. 1997. Information retrieval on the world wide web. *IEEE Internet Computing* 1(5):58–68.

Gunning, D. 2017. Explainable artificial intelligence (XAI). *Defense Advanced Research Projects Agency (DARPA)*.

Herrera, J.; Penas, A.; and Verdejo, F. 2006. Textual entailment recognition based on dependency analysis and WordNet. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment*. Springer. 231–239.

Im, J., and Cho, S. 2017. Distance-based self-attention network for natural language inference. *arXiv preprint arXiv:1712.02047*.

Kotlerman, L.; Dagan, I.; Magnini, B.; and Bentivogli, L. 2015. Textual entailment graphs. *Natural Language Engineering* 21(5):699–724.

Kouylekov, M., and Magnini, B. 2005. Recognizing textual entailment with tree edit distance algorithms. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, 17–20.

Magnini, B.; Zanoli, R.; Dagan, I.; Eichler, K.; Neumann, G.; Noh, T.-G.; Pado, S.; Stern, A.; and Levy, O. 2014. The Excitement Open Platform for textual inferences. In *ACL (System Demonstrations)*, 43–48.

Newman, E.; Stokes, N.; Dunnion, J.; and Carthy, J. 2005. UCD IIRG approach to the textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 53–56.

Parikh, A.; Täckström, O.; Das, D.; and Uszkoreit, J. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2249–2255.

Paul, C.; Rettinger, A.; Mogadala, A.; Knoblock, C. A.; and Szekely, P. 2016. Efficient graph-based document similarity. In *International Semantic Web Conference*, 334–349. Springer.

Pawlik, M., and Augsten, N. 2016. Tree edit distance: Robust and memory-efficient. *Information Systems* 56:157–173.

Pérez, D., and Alfonseca, E. 2005. Application of the Bleu algorithm for recognising textual entailments. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, 9–12.

Runkler, T. A., and Bezdek, J. C. 2003. Web mining with relational clustering. *International Journal of Approximate Reasoning* 32(2-3):217–236.

Sales, J. E.; Souza, L.; Barzegar, S.; Davis, B.; Freitas, A.; and Handschuh, S. 2018. Indra: A word embedding and semantic relatedness server. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Silva, V. S.; Freitas, A.; and Handschuh, S. 2018a. Building a knowledge graph from natural language definitions for interpretable text entailment recognition. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Silva, V. S.; Freitas, A.; and Handschuh, S. 2018b. Recognizing and justifying text entailment through distributional navigation on definition graphs. In *AAAI*.

Silva, V. S.; Handschuh, S.; and Freitas, A. 2016. Categorization of semantic roles for dictionary definitions. In *Cognitive Aspects of the Lexicon (CogALex-V), Workshop at COLING 2016*, 176–184.

Wang, R., and Neumann, G. 2008. An divide-and-conquer strategy for recognizing textual entailment. In *Proceedings of the Text Analysis Conference, Gaithersburg, MD*.

Williams, A.; Nangia, N.; and Bowman, S. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, 1112–1122.

Zanoli, R., and Colombo, S. 2016. A transformation-driven approach for recognizing textual entailment. *Natural Language Engineering* 1–28.

Zhang, K., and Shasha, D. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM journal on computing* 18(6):1245–1262.

Zhang, K.; Chen, E.; Liu, Q.; Liu, C.; and Lv, G. 2017. A context-enriched neural network method for recognizing lexical entailment. In *AAAI*, 3127–3134.