Learning Lexical-Semantic Relations Using Intuitive Cognitive Links

Georgios Balikas[1], Gaël Dias[2], Rumen Moraliyski[3], Houssam Akhmouch[2,4], and Massih-Reza Amini[5]

[1] Kelkoo Group, Grenoble, France
[2] Normandy University, CNRS GREYC, France
[3] Kodar Ltd, Plovdiv, Bulgaria
[4] Credit Agricole Brie Picardie, France
[5] University of Grenoble Alps, CNRS LIG, France

Abstract. Identifying the specific semantic relations between words is crucial for IR and NLP systems. Our goal in this paper is twofold. First, we want to understand whether learning a classifier for one semantic relation (e.g. hypernymy) can gain from concurrently learning another classifier for a cognitively-linked semantic relation (e.g. co-hyponymy). Second, we evaluate how these systems perform where only few labeled examples exist. To answer the first question, we rely on a multi-task neural network architecture, while for the second we use self-learning to evaluate whether semi-supervision improves performance. Our results on two popular datasets as well as a novel dataset proposed in this paper show that concurrent learning of semantic relations consistently benefits performance. On the other hand, we find that semi-supervised learning can be useful depending on the semantic relation. The code and the datasets are available at `https://bit.ly/2Qitasd`.

1 Introduction

The ability to automatically identify lexical-semantic relations is an important issue for Information Retrieval (IR) and Natural Language Processing (NLP) applications such as question answering [12], query expansion [18], or text summarization [13]. Lexical-semantic relations embody a large number of symmetric and asymmetric linguistic phenomena such as synonymy (bike $\leftrightarrow$ bicycle), co-hyponymy (bike $\leftrightarrow$ scooter), hypernymy (bike $\rightarrow$ tandem) or meronymy (bike $\rightarrow$ chain), but more exist [37].

Most approaches focus on a single semantic relation and consist in deciding whether a given relation r holds between a pair of words (x,y). Within this binary classification framework, the vast majority of efforts [36,29,35,25] concentrate on hypernymy, as it is the key organization principle of semantic memory. Other studies can be found on antonymy [26], meronymy [14] and co-hyponymy [38]. Another research direction consists in dealing with several semantic relations simultaneously. This is defined as deciding which semantic relation r_i (if any) holds between a pair of words (x, y). This multi-class problem is challenging as it is known that distinguishing between different semantic relations (e.g. synonymy and hypernymy) is difficult [35].

Recently, [32] showed that symmetric similarity measures that capture synonymy [19] are important features in hypernymy detection. Second, [39] showed that learning term embeddings that take into account co-hyponymy similarity improves hypernymy identification. Such observations imply that learning features that encode one lexical relation can benefit the task of identifying another lexical relation. In this work, we evaluate to what extent this hypothesis holds using four semantic relations: synonymy, co-hyponymy, hypernymy and meronymy. For this purpose, we use multi-task learning where the associated tasks that are learned concurrently are the binary classification problems, which determine the semantic relations between word pairs. Our hypothesis is that if the tasks are cognitively linked, multi-task learning approaches should improve the performance on the tasks as the decision functions are learned concurrently.

In this paper, we also explore the effect of relying on a small amount of labeled data and a larger number of unlabeled data when learning classification models. Indeed, previous works use several (rather small) gold standard datasets of word pairs ignoring the potential of weakly labeled word pairs that can be obtained through selected lexico-syntactic patterns [17] or paraphrase alignments [11]. We argue that such gold-standard datasets may not be available for specific languages or domains. Moreover, human cognition and its generalization capacity is unlikely to rely on the equivalent number of positive examples. Therefore, we propose to use semi-supervised learning methods, both with and without multi-task learning, and evaluate whether they can benefit overall performance amongst all experimented tasks.

Our contributions in this paper are as follows: (1) we show that **multi-task learning** consistently improves the classification performance of semantic relations, (2) we build a **novel dataset** for this specific task that is larger than previously published datasets and will serve the community when developing and evaluating classification methods for semantic relations, and (3) we show that **semi-supervised learning** can benefit performance depending on the used dataset and semantic relation.

2 Related Work

Whether semantic relation identification has been tackled as a binary or a multi-class problem, two main families of approaches have been proposed to capture the semantic links between two words (x, y): pattern-based and distributional. Pattern-based (also called path-based) methods base their decisions on the analysis of the lexico-syntactic patterns (e.g. X *such as* Y) that connect the joint occurrences of x and y. Within this context, earlier works proposed unsupervised [17] and supervised [36] methods to detect hypernymy. However, path-based approaches suffer from sparse coverage and benefit precision over recall. To overcome these limitations, recent two-class studies on hypernymy [35] and antonymy [26], as well as multi-class approaches [34] have been focusing on representing dependency patterns as continuous vectors using long short-term memory networks. Within this context, successful results have been evidenced but [35,26] also show that the combination of pattern-based methods with the distributional approach greatly improves performance.

In distributional methods, the decision whether x is within a semantic relation with y is based on the distributional representation of these words following the distributional hypothesis [16], i.e. on the separate contexts of x and y. Earlier works developed symmetric [11] and asymmetric [20] similarity measures based on discrete representation vectors, followed by numerous supervised learning strategies for a wide range of semantic relations [3,29,38], where word pairs are encoded as the concatenation of the constituent words representations ($\overrightarrow{x} \oplus \overrightarrow{y}$) or their vector difference ($\overrightarrow{x} - \overrightarrow{y}$). More recently, attention has been focusing on identifying semantic relations using neural language embeddings, as such semantic spaces encode linguistic regularities [23]. Within this context, [37] proposed an exhaustive study for a wide range of semantic relations and showed that under suitable supervised training, high performance can be obtained. However, [37] also showed that some relations such as hypernymy are more difficult to model than others. As a consequence, new proposals have appeared that tune word embeddings for this specific task, where hypernyms and hyponyms should be closed to each other in the semantic space [39,25].

In this paper, we propose an attempt to deal with semantic relation identification based on a multi-task strategy, as opposed to previous two-class and multi-class approaches. Our main scope is to analyze whether a link exists between the learning process of related semantic relations. The closest approach to ours is proposed by [2], which develops a multi-task convolutional neural network for multi-class semantic relation classification supported by relatedness classification. As such, it can be seen as a domain adaptation problem. Within the scope of our paper, we aim at studying semantic inter-relationships at a much finer grain and understanding the cognitive links that may exist between synonymy, co-hyponymy, hypernymy and meronymy, that represent a large proportion of any taxonomic structure. For this first attempt, we follow the distributional approach as in [2], although we are aware that improvements may be obtained by the inclusion of pattern-based representations[6]. Moreover, to the best of our knowledge, we propose the first attempt to deal with semantic relation identification based on a semi-supervised approach, thus avoiding the existence of a large number of training examples. As a consequence, we aim at providing a more natural learning framework where only a few labeled examples are initially provided and massively-gathered related word pairs iteratively improve learning.

3 Methodology

3.1 Multi-task with hard parameter sharing

As discussed in [6], not every task combination is beneficial. But, concurrent learning of tasks that have cognitive similarities is often beneficial. We may hypothesize that recognizing the different semantic relations that hold between words can benefit classification models across similar tasks. For instance, learning that *bike* is the hypernym of *mountain bike* should help while classifying *mountain bike* and *tandem bicycle* as co-hyponyms, as it is likely that *tandem bicycle* shares some relation with *bike*. To test this hypothesis, we propose to use a multi-task learning approach. Multi-task learning [8] has been empirically validated and has shown to be effective in a variety of NLP tasks ranging from sentiment analysis to part-of-speech tagging and text parsing [7,6].

[6] This issue is out of the scope of this paper.

The hope is that by jointly learning the decision functions for related tasks, one can achieve better performance. It may be first due to knowledge transfer across tasks that is achieved either in the form of learning more robust representations or due to the use of more data. Second, it has been argued that multi-task learning can act as a regularization process thus preventing from overfitting by requiring competitive performance across different tasks [8].

In this paper, we propose to use a multi-task learning algorithm that relies on hard parameter sharing. Using a simple neural network architecture, our primary objective is to validate our initial hypotheses limiting the effect that choices of architectures and free parameters may have, to the extent of possible. The idea is that the shared parameters (e.g. word representations or weights of some hidden layers) can benefit the performance of all tasks learned concurrently if the tasks are related. In particular, we propose a hard parameter sharing architecture based on a feed-forward neural network (NN) to perform the classification task. The NN architecture is illustrated in Figure 1, based on the overall idea is that there exists a common representation of the input features that can serve to solve all tasks at hand.

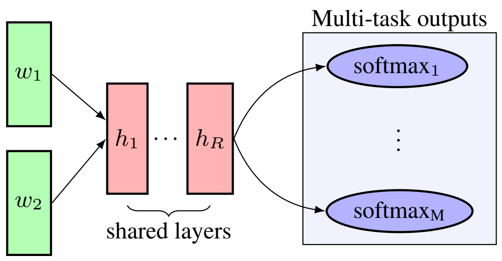

Fig. 1. Feed-forward neural network, where the layers $h_1 \cdots h_R$ are shared across tasks while the output layers softmax$_1 \cdots$ softmax$_M$ are task-dependent.

In this model, all tasks base their decision on the same shared representation[7]. In particular, the input of the network is the concatenation of the word embeddings of the word pairs followed by a series of non-linear hidden layers. Then, a number of softmax layers gives the network predictions. Here, a softmax layer corresponds to a task, and concurrently learning M tasks requires M separate output softmax layers. For example, if the network tries to solve two problems concurrently, like *hypernymy* vs. *random* and *hyponymy* vs. *random*, there will be two independent outputs with separate loss functions, each one solving a dedicated problem. The efficiency of hard parameter sharing architectures relies on the fact that the first layers that are shared are tuned by back-propagating the classification errors of every task. That way, the architecture uses the datasets of all tasks (the two dataset of the two problems in the above example), instead of just one at a time. In Algorithm 1, we detail the training protocol. Note that

[7] We are aware that this architecture can further be improved by additional task-specific inputs, but as a great deal of possible models can be proposed, which deserve intensive research, this issue remains out of the scope of this paper.

the different tasks learned by the NN share the same weights as batches are randomly sampled from their corresponding datasets. Notice that the architecture can be used with different weights for the tasks or can even be tuned with in order to achieve better results on one of the tasks. Automatically learning these weights is an interesting future research direction.

Algorithm 1: Multi-task Training Process

```
Data: Labeled words pairs $\mathcal{L}^i$ for each of the $M$ tasks, batch size b, epochs
epoch = 1 ;
while epoch < epochs do
    for $i = 0;\ i < M;\ i = i + 1$ do
        Randomly select a batch of size $b$ for task $i$ ;
        Update the parameters of the neural network architecture according to the errors
          observed for the batch;
        Calculate the performance on the validation set of task $i$.
    end
end
```

3.2 Semi-supervision via self-learning

Semi-supervised learning approaches perform well in a variety of tasks such as text classification and text summarization [1,9]. As in the supervised learning framework, we assume that we are given access to a set $\mathcal{L} = \{(w_i, w'_i, rel)\}_{i=1}^{i=K}$ that consists of K pairs of words labeled according to the relationship rel. Complementary to that, we also assume to have access to a set of K' words pairs $\mathcal{U} = \{(w_i, w'_i)\}_{i=1}^{i=K'}$ distinct from those of $\mathcal{L}$, and totally unlabeled. The challenge in this setting is to surpass the performance of classification models trained exclusively on $\mathcal{L}$ by using the available data in $\mathcal{U}$. To do so we use self-learning, boosting the training set with confident predictions of an initial classifier. Formally, the underlying idea of self-learning is to train a learner on the set $\mathcal{L}$, and then progressively expand $\mathcal{L}$, by pseudo-labeling N pairs within $\mathcal{U}$, for which the current prediction function is the most confident and adding them to $\mathcal{L}$. This process is repeated until no more pairs are available in $\mathcal{U}$ or, that the performance on a validation set degrades due to the newly-added possibly noisy examples. Algorithm 2 details this process. One point illustrated in Algorithm 2 to be highlighted is that the training set $\mathcal{L}$ is augmented after each iteration of self-learning in a stratified way. In this case, the class distribution of the N pseudo-labeled examples that are added to $\mathcal{L}$ is the same as the class distribution of $\mathcal{L}$. This constraint follows from the independent and identically distributed (i.i.d.) assumption between the $\mathcal{L}$ and $\mathcal{U}$ sets and ensures that the distribution on the classes in the training set does not change as training proceeds. Another point to be mentioned is that the examples that are added to $\mathcal{L}$ may be noisy. Despite the confident predictions of the classifier C, one should expect that some of the instances added are wrongly classified. To reduce the impact of the noise to the training set, we monitor the performance of the classifier using the validation set $\mathcal{V}$ and if the performance degrades the self-learning iteration stops.

Algorithm 2: Self-learning

```
Data: Word pairs: labeled $\mathcal{L}$, unlabeled $\mathcal{U}$, validation $\mathcal{V}$; integer $N$
$\mathcal{L}_0 = \mathcal{L}, \mathcal{U}_0 = \mathcal{U}$ ;
Train classifier $C$ using $\mathcal{L}_0$, $V_0$ : Performance of $C$ on $\mathcal{V}$ ;
Set $t = 0$;
while $Size(\mathcal{U}_t) > 0$ and $V_t \geqslant V_0$ do
    Get probability scores $p$ of $C$ on $\mathcal{U}_t$ ;
    pseudo_labeled$(N) = \arg\max(p)$, stratified wrt $\mathcal{L}_0$ ;
    t = t + 1;
    $\mathcal{L}_t = \mathcal{L}_{t-1} +$ pseudo_labeled ;
    $\mathcal{U}_t = \mathcal{U}_{t-1} -$ pseudo_labeled;
    Retrain $C$ using $\mathcal{L}_t$, $V_t$ : Performance of $C$ on $\mathcal{V}$ ;
end
```

4 Experimental Setups

4.1 Datasets

In order to perform our experiments, we use the ROOT9 dataset[8] [31] that contains 9,600 word pairs, randomly extracted from three well-known datasets: EVALution [33], Lenci/Benotto [5] and BLESS [4]. The word pairs are equally distributed among three classes (hypernymy, co-hyponymy and random) and involve several part-of-speech tags (adjectives, nouns and verbs). Here, we exclusively focus on nouns and keep 1,212 hypernyms, 1,604 co-hyponyms and 549 random pairs that can be represented by GloVe embeddings [28].

In order to include synonymy as a third studied semantic relation, we build the RUMEN dataset[9] that contains 18,978 noun pairs automatically gathered from WordNet 3.0[10] [24] and equally organized amongst three classes (hypernymy, synonymy and random). Note that the words in the pairs are single words and do not contain multi-word expressions. In particular, the RUMEN dataset contains 9,125 word types (i.e. unique nouns) distributed as follows for each semantic relation: 5,054 for hypernymy, 5,201 for synonymy and 6,042 for random. In order to evidence the ambiguity level of the dataset, Table 1 presents the distribution of the types by the number of senses they cover in WordNet. It can be evidenced that while the random category is mainly composed (by construction) of weakly ambiguous nouns, synonymy embodies a large set of polysemous words, while hypernymy contains both weakly polysemous words (usually the hyponym) and more polysemous words (usually the hypernym).

Note that with respect to hypernyms, all noun pairs are randomly selected such that they are not necessarily in direct relation. Exactly 17.2% of the hypernyms are in direct relation, 19.9% have a path length of 2, 20.2% of 3, 16.2% of 4, and 26.5% have a path length superior or equal to 5. Note also that random pairs have as lowest

[8] `https://github.com/esantus/ROOT9`.
[9] Available at `https://bit.ly/2Qitasd`.
[10] `http://wordnetcode.princeton.edu/3.0/`.

common ancestor the root of the hierarchy with a minimum path distance equals to 7[11] between both words so to ensure semantic separateness. On average, each random pair is separated by a path length of 13.2.

Table 1. Distribution of types by number of senses discriminated by semantic category.

# of senses	1	2	3	4	5	6	≥7
RUMEN Hypernymy	24.69%	19.67%	14.21%	9.34%	7.40%	4.82%	19.87%
RUMEN Synonymy	18.46%	18.80%	14.52%	11.04%	8.52%	5.77%	22.89%
RUMEN Random	39.71%	23.28%	12.89%	7.54%	5.05%	3.08%	8.45%

In order to better understand the particularities of the RUMEN dataset, we present the portions of the hierarchy, which are covered by the word pairs in Table 2. To do so, we compute the path length that holds between the root and the highest word of the noun pair in the hierarchy, for all pairs, and calculate the respective distribution.

Table 2. Distribution of pairs by length path from the root discriminated by semantic category.

Path length	0	1	2	3	4	5	6	≥7
RUMEN Hypernymy	5.72%	2.99%	10.22%	27.91%	23.58%	14.86%	9.21%	5.51%
RUMEN Synonymy	0.00%	0.00%	0.08%	1.72%	10.92%	26.14%	26.88%	34.26%
RUMEN Random	0.00%	0.03%	0.69%	5.56%	20.34%	30.07%	22.63%	20.68%

Note that for the synonymy relation, mostly the bottom part (i.e. near the leaves) is covered, while for the hypernymy relation, most pairs have their hypernym in the middle of the hierarchy (levels 3 and 4)[12]. As for the random relation, the pairs are rather uniformly distributed from level 4 to bottom.

Finally, note that for our experiment, we keep 3,375 hypernym, 3,213 synonym and 3,192 random word pairs encoded by GloVe embeddings as many pairs contain unknown words.

4.2 Lexical Splits

Following a classical learning procedure, the datasets must be split into different subsets: train, validation, test and unlabeled in the case of semi-supervision. The standard procedure is random splitting where word pairs are randomly selected without other constraint to form the subsets. However, the authors of [21] point out that using distributional representations in the context of supervised learning tends to perform lexical memorization. In this case, the model mostly learns independent properties of single

[11] This value was set experimentally.

[12] A large number of hypernym pairs contain the root synset "entity", i.e. path length equals to 0.

terms in pairs. For instance, if the training set contains word pairs like ($bike$, $tandem$), ($bike$, *off-roader*) and ($bike$, $velocipede$) tagged as hypernyms, the algorithm may learn that $bike$ is a prototypical hypernym and all new pairs ($bike$, y) may be classified as hypernyms, regardless of the relation that holds between $bike$ and y. To overcome this situation and prevent the model from overfitting by lexical memorization, [21] suggested to split the train and test sets such that each one contains a distinct vocabulary. This procedure is called lexical split. Within the scope of this study, we propose to apply lexical split as defined in [21]. So, lexical repetition exists in the train, validation and the unlabeled subsets, but the test set is exclusive in terms of vocabulary. Table 3 shows the vocabulary and the pairs before and after the lexical splits.

Table 3. Statistics on the datasets and the lexical splits we performed to obtain the train and test subsets. V is the vocabulary size in the original dataset; V_{train} (resp. V_{test}) corresponds to the vocabulary size in the train (resp. test) dataset for the lexical split after removing all words that do not belong to GloVe dictionary. Then, for each lexical relation, we provide the number of word pairs in the train/test datasets.

Dataset	ROOT9	RUMEN	ROOT9+RUMEN	BLESS
Co-hyponyms	939/665	-	1,193/350	1,361/502
Hypernyms	806/486	2,638/737	3,330/1,238	525/218
Meronyms	-	-	-	559/256
Synonyms	-	2,256/957	2,297/1,002	-
Random	339/210	2,227/965	2,630/1,160	2,343/971
V	2,373	9,125	9,779	3,582
$V_{\text{train}}/V_{\text{test}}$	1,423/950	5,475/3,650	5,867/3,912	3,181/2,121

For the specific case of semi-supervised learning, we have further split the pairs dubbed as train so that 60% of them are unlabeled examples. From the remaining 40%, we have randomly selected 30% for validation, resulting in few training examples, which resembles more to a realistic learning scenario where only few positive examples are known. This process is illustrated in Figure 2, with the percentages of the overall dataset. Note that lexical split is not performed between the train, validation and unlabeled subsets[13]. So, while lexical split ensures that the network generalizes to unseen words, it also results in significantly smaller datasets due to the way that these datasets are produced.

4.3 Learning Frameworks

In order to evaluate the effects of our learning strategy, we implement the following baseline systems: (1) Multi-class Logistic Regression using a one-vs-rest approach[14],

[13] All datasets are available at `https://bit.ly/2Qitasd`.

[14] A multi-class model learns to separate between several classes and direct comparison with binary models is not fair. Nevertheless, we report its performance as it highlights the potential of multi-class learning for problems that are cognitively similar.

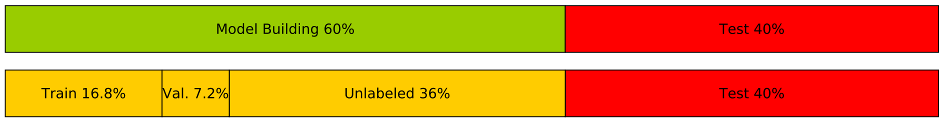

Fig. 2. Illustration of the lexical split of the datasets. The percentages in parentheses correspond to the portions of the original data, used for each purpose.

(2) Logistic Regression that has shown positive results in [32] for hypernymy (i.e. a binary problem), and (3) Feed-forward neural network with two hidden layers of 50 neurons each, which is the direct binary counterpart of our multi-task NN.

For the multi-task learning algorithm, we implemented the architecture shown in Figure 1 using Keras [10]. In particular, we define 2 fully-connected hidden layers (i.e. h_1, h_2, $R = 2$) of 50 neurons each. While the number of hidden layers is a free parameter to tune, we select two hidden layers in advance so that the complexity of the multi-task models are comparable to the neural network baseline. The activation function of the hidden layers is the sigmoid function and the weights of the layers are initialized with a uniform distribution scaled as described in [15]. As for the learning process, we use the Root Mean Square Propagation optimization method with learning rate set to 0.001 and the default value for $\rho = 0.9$. For every task, we use the binary cross-entropy loss function. The network is trained with batches of 32 examples[15]. The word embeddings are initialized with the 300-dimensional representations of GloVe [28].

For the Logistic Regression, we used the implementation of scikit-learn [27]. In particular, a grid search with stratified 3-fold cross validation was used to select the C value in $[0.001, 0.01, 0.1, 1, 10]$.

5 Results

In the following experiments, we report two evaluation measures: Accuracy and Macro-average F_1 measure (MaF_1). Accuracy captures the number of correct predictions over the total predictions, while MaF_1 evaluates how the model performs across the different relations as it uniformly averages the F_1 measures of each relation. In the remaining paragraphs, we comment on three experiments.

In the first experiment, we propose to study the impact of the concurrent learning of co-hyponymy (bike $\leftrightarrow$ scooter) and hypernymy (bike $\rightarrow$ tandem) following the first findings of [39]. For that purpose, we propose to apply our (semi-supervised) multi-task learning strategy over the lexically split ROOT9 dataset using vector concatenation of GloVe [28] as feature representation. Results are illustrated in Table 4. The multi-task paradigm shows that an improved MaF_1 score can be achieved by concurrent learning without semi-supervision achieving a value of 77.3% (maximum value overall). In this case, a 1.1% improvement is obtained over the best baseline (i.e. logistic regression) for hypernymy classification, indeed suggesting that there exists a learning link between

[15] The code is available at `https://bit.ly/2Qitasd`.

hypernymy and co-hyponymy. However, the results for co-hyponymy classification can not compete with a classical supervised strategy using logistic regression. In this case, a 2.1% decrease in MaF$_1$ is evidenced suggesting that the gains for hypernymy classification are not positively balanced by the performance of co-hyponymy. So, we can expect an improvement for hypernymy classification but not for co-hyponymy, suggesting a positive influence of co-hyponymy learning towards hypernymy but not the opposite. Interestingly, the results of the semi-supervised strategy reach comparable figures compared to the multi-task proposal (even superior in some cases), but do not complement each other for the semi-supervised multi-task experiment. In this case, worst results are obtained for both classification tasks suggesting that the multi-task model is not able to correctly generalize from a large number of unlabeled examples, while this is the case for the one-task architecture.

Table 4. Accuracy and MaF$_1$ scores on ROOT9 and RUMEN datasets using GloVe.

		Co-hypo. vs Random		Hyper. vs Random		Average Results	
	Algorithm	Accuracy	MaF$_1$	Accuracy	MaF$_1$	Accuracy	MaF$_1$
ROOT9	Multi-class Logistic Regression	0.740	0.500	0.781	0.507	0.760	0.500
	Logistic Regression	**0.893**	0.854	0.814	0.762	**0.854**	0.808
	NN Baseline	0.890	0.851	0.803	0.748	0.847	0.800
	Self-learning	0.869	**0.859**	0.816	0.772	0.843	**0.815**
	Multi-task learning	0.882	0.833	**0.818**	**0.773**	0.850	0.803
	Multi-task learning + Self-learning	0.854	0.811	0.810	0.767	0.832	0.789
		Syn. vs Random		Hyper. vs Random		Average Results	
	Algorithm	Accuracy	MaF$_1$	Accuracy	MaF$_1$	Accuracy	MaF$_1$
RUMEN	Multi-class Logistic Regression	0.600	0.430	0.620	0.467	0.610	0.448
	Logistic Regression	0.628	0.628	0.711	0.706	0.670	0.667
	NN Baseline	0.679	0.678	0.752	0.748	0.716	0.713
	Self-learning	0.686	0.685	0.757	0.754	0.722	0.720
	Multi-task learning	0.706	0.700	0.755	0.750	0.731	0.725
	Multi-task learning + Self-learning	**0.708**	**0.708**	**0.760**	**0.755**	**0.734**	**0.732**

In the second experiment, we propose to study the impact of the concurrent learning of synonymy (bike $\leftrightarrow$ bicycle) and hypernymy following the experiments of [32] which suggest that symmetric similarity measures (usually tuned to detect synonymy [19]) improve hypernymy classification. For that purpose, we propose to apply the same models over the lexically split RUMEN dataset. Results are illustrated in Table 4. The best configuration is the combination of multi-task learning with self-learning achieving maximum accuracy and MaF$_1$ scores for both tasks. The improvement equals to 0.7% in terms of MaF$_1$ for hypernymy and reaches 3% in terms of MaF$_1$ for synonymy when compared to the best baseline (i.e. neural network). The overall average improvement (i.e. both tasks combined[16]) reaches 1.8% for accuracy and 1.9% for MaF$_1$ over the best baseline. So, these results tend to suggest that synonymy identification may positively be impacted by the concurrent learning of hypernymy and vice versa (although to a less extent). In fact, these results consistently build upon the positive results of the

[16] Column 3 of Table 4.

multi-task strategy without semi-supervision and the self-learning approach alone that both improve over the best baseline results. Nevertheless, the improvement obtained by combining multi-task learning and semi-supervision is negligible compared to multi-task alone. Note also that the results obtained over the RUMEN dataset by the baseline classifiers are lower than the ones reached over ROOT9 for hypernymy, certainly due to the complexity of the datasets themselves. So, we may hypothesize that the multi-task strategy plays an important role by acting as a regularization process and helping in solving learning ambiguities, and reaches improved results over the two-task classifiers.

In the **third experiment**, we propose to study the impact of the concurrent learning of co-hyponymy, synonymy and hypernymy together. The idea is to understand the inter-relation between these three semantic relations that form the backbone of any taxonomic structure. For that purpose, we propose to apply the models proposed in this paper over the lexically split ROOT9+RUMEN dataset[17]. Results are illustrated in Table 5. The best configuration for all the tasks combined (i.e. co-hyponymy, synonymy and hypernymy) is multi-task learning without semi-supervision. Overall, improvements up to 1.4% in terms of accuracy and 2% in terms of MaF_1 can be reached over the best baseline (i.e. neural network). In particular, the MaF_1 score increases 4.4% with the multi-task strategy without self-learning for co-hyponymy, while the best result for synonymy is obtained by the semi-supervised multi-task strategy with an improvement of 1.1% MaF_1 score. The best configuration for hypernymy is evidenced by self-learning alone, closely followed by the multi-task model, reaching improvements in MaF_1 scores of 1.7% (resp. 1%) for self-learning (resp. multi-task learning). Comparatively to the first experiment, both learning paradigms (i.e. semi-supervision and multi-task) tend to produce competitive results alone, both exceeding results of the best baseline. However, the multi-task model hardly generalizes from the set of unlabeled examples, being synonymy the only exception. Finally, note that co-hyponymy seems to be the simplest task to solve, while synonymy is the most difficult one, over all experiments.

In the **fourth experiment**, We now study the meronymy relation (bike $\rightarrow$ chain) into a multi-task environment, as it has traditionally been studied together with hypernymy [14]. The overall idea is to verify whether meronymy can benefit from the concurrent learning of the backbone semantic relations that form knowledge bases. For that purpose, we apply our learning models over the lexically split BLESS dataset [4] that includes three semantic relations: co-hyponymy, hypernymy and meronymy. The details of the lexical split is presented in Table 3 and note that the BLESS dataset has been processed in the exact same way as ROOT9 and RUMEN, i.e. retaining only noun categories and word pairs that can be represented by the GloVe semantic space. Results are presented in Table 5. The best configuration over the three tasks combined is obtained by the semi-supervised multi-task strategy with a MaF_1 score equals to 80.3%, thus improving 1.2% over the best baseline (i.e. neural network). In particular, we can notice that the most important improvement is obtained for the meronymy relation that reaches 73.3% for MaF_1 and 76.4% for accuracy with the multi-task model without semi-supervision. In this particular case, the improvement is up to 2.6% in accuracy and 2.4% in MaF_1 over the neural network baseline. For co-hyponymy (resp. hypernymy), best re-

[17] Note that due to the lexical split process, results can not directly be compared to the ones obtained over ROOT9 or RUMEN.

Table 5. Accuracy and MaF$_1$ scores on ROOT9+RUMEN and BLESS datasets using GloVe.

		Co-hypo. vs Random		Hyper. vs Random		Syn. vs Random		Average Results	
	System	Accuracy	MaF$_1$	Accuracy	MaF$_1$	Accuracy	MaF$_1$	Accuracy	MaF$_1$
ROOT+RUMEN	Multi-class Log. Reg.	0.606	0.370	0.560	0.320	0.500	0.280	0.555	0.323
	Logistic Regression	0.909	0.872	0.669	0.669	0.634	0.632	0.737	0.724
	NN Baseline	0.914	0.875	0.712	0.712	0.663	0.659	0.763	0.748
	Self-learning	0.928	0.900	**0.729**	**0.729**	0.668	0.665	0.775	0.765
	Multi-task learning	**0.943**	**0.919**	0.723	0.722	0.666	0.664	**0.777**	**0.768**
	Multi-task learning + Self.	0.939	0.911	0.711	0.711	**0.672**	**0.670**	0.774	0.764

		Co-hypo. vs Random		Hyper. vs Random		Mero. vs Random		Average Results	
	System	Accuracy	MaF$_1$	Accuracy	MaF$_1$	Accuracy	MaF$_1$	Accuracy	MaF$_1$
BLESS	Multi-class Log. Reg.	0.760	0.408	0.720	0.355	0.722	0.362	0.734	0.375
	Logistic Regression	0.845	0.830	0.888	0.794	0.748	0.723	0.827	0.782
	NN Baseline	0.870	0.855	0.892	0.809	0.738	0.709	0.833	0.791
	Self-learning	0.877	**0.863**	0.900	0.807	0.749	0.723	0.842	0.798
	Multi-task learning	0.866	0.847	**0.903**	**0.816**	**0.764**	**0.733**	**0.844**	0.799
	Multi-task learning + Self.	**0.878**	**0.863**	0.900	0.813	0.754	**0.733**	**0.844**	**0.803**

sults are obtained by multi-task with semi-supervision (resp. without semi-supervision), but show limited improvements over the best baseline, suggesting that meronymy gains more in performance from the concurrent learning of co-hyponymy and hypernymy than the contrary, although improvements are obtained in all cases. Comparatively to the other experiments, we also notice that although the self-learning algorithm and the multi-task framework without semi-supervision perform well alone, the combination of both strategies does not necessary lead to the best results overall, suggesting that the present architecture can be improved by the massive extraction of unlabeled examples.

6 Conclusions

In this paper, we proposed to study the concurrent learning of cognitively-linked semantic relations (co-hyponymy, hypernymy, synonymy and meronymy) using **semi-supervised** and **multi-task learning**. Our results show that concurrent learning leads to improvements in most tested situations and datasets, including the **newly-built dataset** called RUMEN. In particular, results show that hypernymy can gain from co-hyponymy, synonymy from hypernymy, co-hyponymy from both hypernymy and synonymy, and meronymy from both co-hyponymy and hypernymy. Moreover, it is interesting to notice that in three cases out of four, the improvement achieved by the multi-task strategy is obtained for the most difficult task to handle. Nevertheless, there still exists a great margin for improvement. First, we intend to propose new multi-task architectures that include task-specific features similarly to [22] as well as LSTM path-based features as in [35]. Second, we expect to build on new semi-supervised multi-task architectures such as Tri-training [30] to positively combine semi-supervision and multi-task learning as their combination is currently not beneficial in a vast majority of cases. Third, we intend to massively gather unlabeled examples by lexico-syntactic patterns [17] or by paraphrase alignment [11] instead of simulating such a behaviour, as we do currently. Finally, we plan to test all our configurations in “noisy” situations as proposed in [37].

References

1. Amini, M., Laviolette, F., Usunier, N.: A transductive bound for the voted classifier with an application to semi-supervised learning. In: 22nd Annual Conference on Neural Information Processing Systems (NIPS). pp. 65–72 (2008)
2. Attia, M., Maharjan, S., Samih, Y., Kallmeyer, L., Solorio, T.: Cogalex-v shared task: Ghhh - detecting semantic relations via word embeddings. In: Workshop on Cognitive Aspects of the Lexicon. pp. 86–91 (2016)
3. Baroni, M., Bernardi, R., Do, N.Q., chieh Shan, C.: Entailment above the word level in distributional semantics. In: 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL). pp. 23–32 (2012)
4. Baroni, M., Lenci, A.: How we blessed distributional semantic evaluation. In: Workshop on Geometrical Models of Natural Language Semantics (GEMS) associated to Conference on Empirical Methods on Natural Language Processing (EMNLP). pp. 1–10 (2011)
5. Benotto, G.: Distributional Models for Semantic Relations: A Sudy on Hyponymy and Antonymy. Ph.D. thesis, University of Pisa (2015)
6. Bingel, J., Søgaard, A.: Identifying beneficial task relations for multi-task learning in deep neural networks. arXiv preprint arXiv:1702.08303 (2017)
7. Braud, C., Plank, B., Søgaard, A.: Multi-view and multi-task training of RST discourse parsers. In: 26th International Conference on Computational Linguistics (COLING). pp. 1903–1913 (2016)
8. Caruana, R.: Multitask learning. In: Learning to learn, pp. 95–133. Springer (1998)
9. Chapelle, O., Scholkopf, B., Zien, A.: Semi-supervised learning. IEEE Transactions on Neural Networks **20**(3), 542–542 (2009)
10. Chollet, F.: Keras. `https://keras.io` (2015)
11. Dias, G., Moraliyski, R., ao Paulo Cordeiro, J., Doucet, A., Ahonen-Myka, H.: Automatic discovery of word semantic relations using paraphrase alignment and distributional lexical semantics analysis. Natural Language Engineering **16**(4), 439–467 (2010)
12. Dong, L., Mallinson, J., Reddy, S., Lapata, M.: Learning to paraphrase for question answering. In: Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 886–897 (2017)
13. Gambhir, M., Gupta, V.: Recent automatic text summarization techniques: a survey. Artificial Intelligence Review **47**(1), 1–66 (2017)
14. Glavas, G., Ponzetto, S.P.: Dual tensor model for detecting asymmetric lexico-semantic relations. In: Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1758–1768 (2017)
15. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: 13th International Conference on Artificial Intelligence and Statistics. pp. 249–256 (2010)
16. Harris, Z.S.: Distributional structure. Word **10**(2-3), 146–162 (1954)
17. Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In: 14th Conference on Computational Linguistics (COLING). pp. 539–545 (1992)
18. Kathuria, N., Mittal, K., Chhabra, A.: A comprehensive survey on query expansion techniques, their issues and challenges. International Journal of Computer Applications **168**(12) (2017)
19. Kiela, D., Hill, F., Clark, S.: Specializing word embeddings for similarity or relatedness. In: Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 2044–2048 (2015)
20. Kotlerman, L., Dagan, I., Szpektor, I., Zhitomirsky-Geffet, M.: Directional distributional similarity for lexical inference. Natural Language Engineering **16**(4), 359–389 (2010)

21. Levy, O., Remus, S., Biemann, C., Dagan, I.: Do supervised distributional methods really learn lexical inference relations? In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 970–976 (2015)
22. Liu, P., Qiu, X., Huang, X.: Adversarial multi-task learning for text classification. In: 55th Annual Meeting of the Association for Computational Linguistics (ACL) (2017)
23. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR **abs/1301.3781** (2013)
24. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to wordnet: An on-line lexical database. International Journal of Lexicography **3**(4), 235–244 (January 1990)
25. Nguyen, K.A., Köper, M., Schulte im Walde, S., Vu, N.T.: Hierarchical embeddings for hypernymy detection and directionality. In: Conference on Empirical Methods in Natural Language Processing. pp. 233–243 (2017)
26. Nguyen, K.A., Schulte im Walde, S., Vu, N.T.: Distinguishing antonyms and synonyms in a pattern-based neural network. In: 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL). pp. 76–85 (2017)
27. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. Journal of machine learning research **12**(Oct), 2825–2830 (2011)
28. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Conference on Empirical Methods on Natural Language Processing (EMNLP). pp. 1532–1543 (2014)
29. Roller, S., Erk, K., Boleda, G.: Inclusive yet selective: Supervised distributional hypernymy detection. In: 25th International Conference on Computational Linguistics (COLING). pp. 1025–1036 (2014)
30. Ruder, S., Plank, B.: Strong baselines for neural semi-supervised learning under domain shift. In: 56th Annual Meeting of the Association for Computational Linguistics (ACL) (2018)
31. Santus, E., Lenci, A., Chiu, T., Lu, Q., Huang, C.: Nine features in a random forest to learn taxonomical semantic relations. In: 10th International Conference on Language Resources and Evaluation. pp. 4557–4564 (2016)
32. Santus, E., Shwartz, V., Schlechtweg, D.: Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. In: 15th Conference of the European Chapter of the Association for Computational Linguistics. pp. 65–75 (2017)
33. Santus, E., Yung, F., Lenci, A., Huang, C.R.: Evalution 1.0: an evolving semantic dataset for training and evaluation of distributional semantic models. In: 4th Workshop on Linked Data in Linguistics (LDL) associated to Association for Computational Linguistics and Asian Federation of Natural Language Processing (ACL-IJCNLP). pp. 64–69 (2015)
34. Shwartz, V., Dagan, I.: Cogalex-v shared task: Lexnet - integrated path-based and distributional method for the identification of semantic relations. CoRR **abs/1610.08694** (2016)
35. Shwartz, V., Goldberg, Y., Dagan, I.: Improving hypernymy detection with an integrated path-based and distributional method. In: 54th Annual Meeting of the Association for Computational Linguistics. pp. 2389–2398 (2016)
36. Snow, R., Jurafsky, D., Ng, A.Y.: Learning syntactic patterns for automatic hypernym discovery. In: 17th International Conference on Neural Information Processing Systems (NIPS). pp. 1297–1304 (2004)
37. Vylomova, E., Rimell, L., Cohn, T., Baldwin, T.: Take and took, gaggle and goose, book and read: Evaluating the utility of vector differences for lexical relation learning. In: 54th Annual Meeting of the Association for Computational Linguistics. pp. 1671–1682 (2016)

38. Weeds, J., Clarke, D., Reffin, J., Weir, D.J., Keller, B.: Learning to distinguish hypernyms and co-hyponyms. In: 5th International Conference on Computational Linguistics (COLING). pp. 2249–2259 (2014)
39. Yu, Z., Wang, H., Lin, X., Wang, M.: Learning term embeddings for hypernymy identification. In: 24th International Joint Conference on Artificial Intelligence. pp. 1390–1397 (2015)