Concurrent Learning of Semantic Relations

Anonymous EMNLP submission

Abstract

Discovering whether words are semantically related and identifying the specific semantic relation that holds between them is of crucial importance for NLP as it is essential for tasks like query expansion in IR. Within this context, different methodologies have been proposed that either exclusively focus on a single lexical relation (e.g. hypernymy vs. random) or learn specific classifiers capable of identifying multiple semantic relations (e.g. hypernymy vs. synonymy vs. random). In this paper, we propose another way to look at the problem that relies on the multi-task learning paradigm. In particular, we want to study whether the learning process of a given semantic relation (e.g. hypernymy) can be improved by the concurrent learning of another semantic relation (e.g. co-hyponymy). Within this context, we particularly examine the benefits of semi-supervised learning where the training of a prediction function is performed over few labeled data jointly with many unlabeled ones. Preliminary results based on simple learning strategies and state-of-the-art distributional feature representations show that concurrent learning can lead to improvements in a vast majority of tested situations.

1 Introduction

The ability to automatically identify semantic relations is an important issue for Information Retrieval (IR) and Natural Language Processing (NLP) applications such as question answering (Dong et al., 2017), query expansion (Kathuria et al., 2017), or text summarization (Gambhir and Gupta, 2017). Semantic relations embody a large number of symmetric and asymmetric linguistic phenomena such as synonymy (bike ↔ bicycle), co-hyponymy (bike ↔ scooter), hypernymy (bike → tandem) or meronymy (bike → chain), but more can be enumerated (Vylomova et al., 2016).

Most approaches focus on modeling a single semantic relation and consist in deciding whether a given relation r holds between a pair of words (x,y) or not. Within this context, the vast majority of efforts (Snow et al., 2004; Roller et al., 2014; Shwartz et al., 2016; Nguyen et al., 2017a) concentrate on hypernymy which is the key organization principle of semantic memory, but studies exist on antonymy (Nguyen et al., 2017b), meronymy (Glavas and Ponzetto, 2017) and co-hyponymy (Weeds et al., 2014). Another research direction consists in dealing with multiple semantic relations at a time and can be 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 (Shwartz et al., 2016). Within the CogALex-V shared task[1] which aims at tackling synonymy, antonymy, hypernymy and meronymy as a multi-class problem, best performing systems are proposed by (Shwartz and Dagan, 2016) and (Attia et al., 2016).

In this paper, we propose another way to look at the problem of semantic relation identification based on the following findings. First, symmetric similarity measures, which capture synonymy (Kiela et al., 2015) have shown to play an important role in hypernymy detection (Santus et al., 2017). Second, (Yu et al., 2015) show that learning term embeddings that take into account co-hyponymy similarity improves supervised hypernymy identification. As a consequence, we propose to study whether the learning process of a given semantic relation can be improved by the concurrent learning of another relation, where semantic relations are either synonymy, co-hyponymy or hypernymy. For that purpose, we

[1] https://sites.google.com/site/cogalex2016/home/shared-task

propose a multi-task learning strategy using a hard parameter sharing neural network model that takes as input a learning word pair (x, y) encoded as a feature vector representing the concatenation[2] of the respective word embeddings of x and y noted $\overrightarrow{x} \oplus \overrightarrow{y}$. The intuition behind our experiments is that if the tasks are correlated, the neural network should improve its generalization ability by taking into account the shared information.

In parallel, we propose to study the generalization ability of the multi-task learning model based on a limited set of labeled word pairs and a large number of unlabeled samples, i.e. following a semi-supervised paradigm. As far as we know, most related works rely on the existence of a huge number of validated word pairs present in knowledge databases (e.g. WordNet) to perform the supervised learning process. However, such resources may not be available for specific languages or domains. Moreover, it is unlikely that human cognition and its generalization capacity rely on the equivalent number of positive examples. As such, semi-supervised learning proposes a much more interesting framework where unlabeled word pairs can massively[3] be obtained through selected lexico-syntactic patterns (Hearst, 1992) or paraphrase alignments (Dias et al., 2010). To test our hypotheses, we propose a self-learning strategy where confidently tagged unlabeled word pairs are iteratively added to the labeled dataset.

Preliminary results based on simple (semi-supervised) multi-task learning models with state-of-the-art word pairs representations (i.e. concatenation of GloVe (Pennington et al., 2014) word embeddings) over the gold standard dataset ROOT9 (Santus et al., 2016) and the RUMEN dataset proposed in this paper show that classification improvements can be obtained for a wide range of tested configurations.

2 Related Work

Whether semantic relation identification has been tackled as a one-class or a multi-class problem, two main approaches have been addressed to capture the semantic links between two words (x, y): pattern-based and distributional. Pattern-based (a.k.a. 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 have been proposed by (Hearst, 1992) (unsupervised) and (Snow et al., 2004) (supervised) to detect hypernymy. However, this approach suffers from sparse coverage and benefits precision over recall. To overcome these limitations, recent one-class studies on hypernymy (Shwartz et al., 2016) and antonymy (Nguyen et al., 2017b), as well as multi-class approaches (Shwartz and Dagan, 2016) have been focusing on representing dependency patterns as continuous vectors using long short-term memory (LSTM) networks. Within this context, successful results have been evidenced but (Shwartz et al., 2016; Nguyen et al., 2017b) 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 (Harris, 1954), i.e. on the separate contexts of x and y. Earlier works developed symmetric (Dias et al., 2010) and asymmetric (Kotlerman et al., 2010) similarity measures based on discrete representation vectors, followed by numerous supervised learning strategies for a wide range of semantic relations (Baroni et al., 2012; Roller et al., 2014; Weeds et al., 2014), 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 (Mikolov et al., 2013). Within this context, (Vylomova et al., 2016) proposed an exhaustive study for a wide range of semantic relations and showed that under suitable supervised training, high performance can be obtained. However, (Vylomova et al., 2016) 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 (Yu et al., 2015; Nguyen et al., 2017a).

In this paper, we propose an attempt to deal with semantic relation identification based on a multi-task strategy, as opposed to previous one-class and

[2] Best configuration reported in (Shwartz et al., 2016) for standard non path-based supervised learning.

[3] Even though with some error rate.

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 (Attia et al., 2016), 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 and hypernymy, that form the backbone of any taxonomic structure. For this first attempt, we follow the distributional approach as in (Attia et al., 2016), although we are aware that improvements may be obtained by the inclusion of pattern-based representations[4]. Moreover, we propose the first attempt[5] to deal with semantic relation identification based on a semi-supervised approach, thus avoiding the pre-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 (Bingel and Søgaard, 2017), 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 (Caruana, 1998) has empirically been validated and has shown to be effective in a variety of NLP tasks ranging from sentiment analysis, part-of-speech tagging and text parsing (Braud et al., 2016; Bingel and Søgaard, 2017). 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 (Caruana, 1998).

In this paper, we propose to use a multi-task learning algorithm that relies on hard parameter sharing[6]. 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 step. The NN architecture is illustrated in Figure 1.

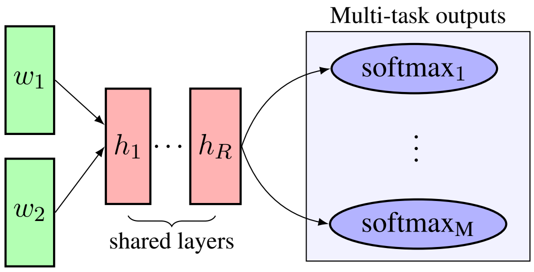

Figure 1: The multi-task learning architecture is built on top of a feed-forward neural network, where the layers $h_1 \cdots h_R$ are shared across tasks while the output layers softmax$_1 \cdots$ softmax$_M$ are task-dependent.

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. 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, instead of just one at a time. In Algorithm 1, we detail the training protocol. Note that the different tasks learned by the NN share the same weights as batches are randomly sampled from their corresponding datasets[7].

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

[5] As far as we know.

[6] We use a simple architecture as our primary objective is to validate our initial hypotheses and not necessarily focus on overall performance.

[7] Automatically learning different weights for the tasks and self-adjusting them for the sake of overall performance is a possible 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
```

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
```

3.2 Semi-supervision via self-learning

Semi-supervised learning approaches have shown to perform well in a variety of tasks such as text classification and text summarization (Amini and Gallinari, 2002; Chapelle et al., 2009). 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 $\mathcal{U}$. While several methods have been proposed, we opt to use self-learning as the semi-supervised algorithm, as it is one of the simplest approaches[8]. The central idea behind 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.

4 Experimental Setups

In this section, we present the experimental setups of our study so that our results can easily be reproduced. In particular, we detail the learning frameworks, and the datasets and their splits.

4.1 Learning Frameworks

In order to evaluate the effects of our learning strategy, we implement the following baseline systems: (1) Majority Baseline; (2) Logistic Regression that has shown positive results in (Santus et al., 2017), and (3) Feed-forward neural network with two hidden layers of 50 neurons each, which is the direct one-task counterpart of our NN multi-task architecture. For the multi-task learning algorithm, we implemented the architecture shown in Figure 1 using Keras (Chollet, 2015). 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 word embeddings are initialized with the 300-

[8] The main objective of this paper is to verify the benefits of semi-supervised multi-task learning for our task and not to tune complex solutions that can reach high performance. This remains a future work.

dimensional representations of GloVe (Pennington et al., 2014). 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 (Glorot and Bengio, 2010). As for the learning process, we use the Root Mean Square Propagation (RMSprop) 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[9]. For the Logistic Regression, we used the implementation of scikit-learn (Pedregosa et al., 2011).

4.2 Datasets and Splits

In order to perform our experiments, we use the ROOT9 dataset[10] (Santus et al., 2016) that contains 9,600 word pairs, randomly extracted from three well-known datasets: EVALution (Santus et al., 2015), Lenci/Benotto (Benotto, 2015) and BLESS (Baroni and Lenci, 2011). 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 (Pennington et al., 2014).

In order to include synonymy as a third studied semantic relation, we build the RUMEN dataset[11] that contains 18,978 word pairs equally organized amongst three classes (hypernymy, synonymy and random). By doing so, we expect to move to more challenging settings and overcome the limitations of only using hypernyms and co-hyponyms (Camacho-Collados, 2017). In RUMEN, all noun pairs are randomly selected based on WordNet 3.0[12] (Miller et al., 1990) such that hypernyms are not necessarily in direct relation and random pairs have as most common parent the root of the hierarchy with a minimum path distance equals to 7[13] to ensure semantic separateness. Finally, we keep 3,375 hypernym, 3,213 synonym and 3,192 random word pairs encoded by GloVe embeddings.

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, (Levy et al., 2015) 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 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, (Levy et al., 2015) 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 (Levy et al., 2015). So, lexical repetition exists in the train, validation and the unlabeled subsets, but the test set is exclusive in terms of vocabulary. Table 1 shows the vocabulary and the pairs before and after the lexical splits. In our experiments, 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. 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. All subsets are available for replicability[14].

5 Results

In the experiments that follow, 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 averages the F_1 measures of each relation without weighting the number of examples in each case. In the remaining of this section, we comment on three experiments.

[9] Note that the code will be available for research purposes upon acceptance.

[10] https://github.com/esantus/ROOT9

[11] http://anonymous (freely available)

[12] http://wordnetcode.princeton.edu/3.0/

[13] This value was set experimentally.

[14] http://anonymous (freely available)

Dataset	Pairs	V	$V_{\text{train}}/V_{\text{test}}$	Co-hyp.	Hypernyms	Mero.	Synonyms	Random
ROOT9	6,747	2,373	1,423/950	939/665	806/486	N/A	N/A	339/210
RUMEN	18,979	9,125	5,475/3,650	N/A	2,638/737	N/A	2,256/957	2,227/965
ROOT9+RUMEN	25,726	9,779	5,867/3,912	1,193/350	3,330/1,238	N/A	2,297/1,002	2,630/1,160
BLESS	14,547	3,582	3,181/2,121	1,361/502	525/218	559/256	N/A	2,343/971

Table 1: 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 relations, we provide the number of word pairs in the train/test datasets. During pre-processing, the train subset has been further split into train, validation and unlabeled sets as explained in section 4.

		Co-hypo. vs Random		Hyper. vs Random		Average Results	
	Algorithm	Accuracy	MaF$_1$	Accuracy	MaF$_1$	Accuracy	MaF$_1$
ROOT9	Majority Baseline	0.760	0.431	0.698	0.411	0.729	0.421
	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**
	Multitask learning	0.882	0.833	**0.818**	**0.773**	0.850	0.803
	Multitask 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	Majority Baseline	0.496	0.331	0.432	0.301	0.464	0.316
	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
	Multitask learning	0.706	0.700	0.755	0.750	0.731	0.725
	Multitask learning + Self-learning	**0.708**	**0.708**	**0.760**	**0.755**	**0.734**	**0.732**

Table 2: Accuracy and Macro F$_1$ scores on ROOT9 and RUMEN datasets for GloVe semantic space.

In the **first experiment**, we propose to study the impact of the concurrent learning of co-hyponymy (bike ↔ scooter) and hypernymy (bike → tandem) following the first findings of (Yu et al., 2015). 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 (Pennington et al., 2014) as feature representation. Results are illustrated in Table 2. 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 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 in a multi-task environment, 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.

In the **second experiment**, we propose to study the impact of the concurrent learning of synonymy (bike ↔ bicycle) and hypernymy following the experiments of (Santus et al., 2017) which suggest that symmetric similarity measures (usually tuned to detect synonymy (Kiela et al., 2015)) improve hypernymy classification. For that purpose, we propose to apply the same models over the lexically split RUMEN dataset. Results are illustrated in Table 2. 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[15]) 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 greatly 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 multi-task strategy without semi-supervision and the self-learning approach alone that both improve over the best baseline results. Note 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 one-task classifiers.

In the **third experiment**, we propose to study the impact of the concurrent learning of co-hyponymy, synonymy and hypernymy all 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[16]. Results are illustrated in Table 3. 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.

6 Studying Meronymy

In this section, we study the introduction of the meronymy relation (bike → chain) into a multi-task environment, as it has traditionally been studied together with hypernymy (Glavas and Ponzetto, 2017). The overall idea is to verify whether the meronymy semantic relation 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 (Baroni and Lenci, 2011) that includes three semantic relations: co-hyponymy, hypernymy and meronymy. The details of the lexical split is presented in Table 1 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 3. 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 results 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 al-

[15] Column 3 of Table 2.

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

		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	Majority Baseline	0.768	0.434	0.516	0.340	0.536	0.350	0.607	0.375
	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
	Multitask learning	**0.943**	**0.919**	0.723	0.722	0.666	0.664	**0.777**	**0.768**
	Multitask 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	Majority Baseline	0.660	0.397	0.816	0.449	0.653	0.395	0.710	0.414
	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
	Multitask learning	0.866	0.847	**0.903**	**0.816**	**0.764**	**0.733**	**0.844**	0.799
	Multitask learning + Self.	**0.878**	**0.863**	0.900	0.813	0.754	**0.733**	**0.844**	**0.803**

Table 3: Accuracy and Macro F$_1$ scores on ROOT9+RUMEN and BLESS datasets for GloVe semantic space.

though 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 to positively gain from the massive extraction of unlabeled examples.

7 Conclusions and Discussion

In this paper, we proposed to study the concurrent learning of semantic relations (co-hyponymy, hypernymy, synonymy and meronymy) using simple learning strategies (self-learning and hard parameter sharing multi-task learning) and state-of-the-art continuous distributional representations (concatenation of GloVe embeddings). The idea was to verify if the concurrent learning of these relations that share some cognitive similarities could be beneficial, without necessarily focusing on overall performance. Obtained results show that concurrent learning can lead to improvements in a vast majority of tested situations. In particular, we have shown that within this experimental framework, 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 obtained by the multi-task strategy is obtained for the most difficult task to handle, thus suggesting the benefits of concurrent learning. Based on these preliminary findings, a vast amount of improvements can now be introduced into the framework to increase overall performance. With respect to the input features, we intend to study the potential benefits from dedicated embeddings such as hypervec (Nguyen et al., 2017a), knowledge graphs embeddings (Speer et al., 2017) and dual embeddings (Nalisnick et al., 2016). Moreover, we deeply believe that the LSTM path-based features introduced in (Shwartz et al., 2016) and some well-defined word pairs similarity measures (Santus et al., 2017) can lead to classification improvements by complementing the information present in distributional semantic spaces. With respect to the learning framework, some clear efforts must be performed. Indeed, the current combination of self-learning and hard parameter sharing multi-task learning is not beneficial in a vast majority of cases suggesting the proposition of new architectures following the ideas of Tri-training (Ruder and Plank, 2018). Moreover, it is clear that more complex architectures such as convolutional neural networks may improve the learning process as it is proposed in (Attia et al., 2016) for similar tasks. As semi-supervision is concerned, we intend to study a more realistic situation where unlabeled examples are massively gathered by lexico-syntactic patterns (Hearst, 1992) or by paraphrase alignment (Dias et al., 2010) over huge raw text corpora. Indeed, here, semi-supervision is just simulated by fractioning the existing datasets. Finally, we plan to difficult the original task by including the detection of the direction of the asymmetric semantic relations, testing the combination of more closely related semantic relations and including noisy pairs as in (Vylomova et al., 2016).

References

Massih-Reza Amini and Patrick Gallinari. 2002. The use of unlabeled data to improve supervised learning for text summarization. In *25th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 105–112.

Mohammed Attia, Suraj Maharjan, Younes Samih, Laura Kallmeyer, and Thamar Solorio. 2016. Cogalex-v shared task: Ghhh - detecting semantic relations via word embeddings. In *Workshop on Cognitive Aspects of the Lexicon*, pages 86–91.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung chieh Shan. 2012. Entailment above the word level in distributional semantics. In *13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 23–32.

Marco Baroni and Alessandro Lenci. 2011. 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)*, pages 1–10.

Giulia Benotto. 2015. *Distributional Models for Semantic Relations: A Sudy on Hyponymy and Antonymy*. Ph.D. thesis, University of Pisa.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. *arXiv preprint arXiv:1702.08303*.

Chloé Braud, Barbara Plank, and Anders Søgaard. 2016. Multi-view and multi-task training of RST discourse parsers. In *26th International Conference on Computational Linguistics (COLING)*, pages 1903–1913.

Jose Camacho-Collados. 2017. Why we have switched from building full-fledged taxonomies to simply detecting hypernymy relations. *arXiv preprint arXiv:1703.04178*.

Rich Caruana. 1998. Multitask learning. In *Learning to learn*, pages 95–133. Springer.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. 2009. Semi-supervised learning. *IEEE Transactions on Neural Networks*, 20(3):542–542.

François Chollet. 2015. Keras. https://keras.io.

Gaël Dias, Rumen Moraliyski, Jo ao Paulo Cordeiro, Antoine Doucet, and Helena Ahonen-Myka. 2010. Automatic discovery of word semantic relations using paraphrase alignment and distributional lexical semantics analysis. *Natural Language Engineering*, 16(4):439–467.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 886–897.

Mahak Gambhir and Vishal Gupta. 2017. Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66.

Goran Glavas and Simone Paolo Ponzetto. 2017. Dual tensor model for detecting asymmetric lexico-semantic relations. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1758–1768.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *13th International Conference on Artificial Intelligence and Statistics (AISTAT)*, pages 249–256.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *14th Conference on Computational Linguistics (COLING)*, pages 539–545.

Neha Kathuria, Kanika Mittal, and Anusha Chhabra. 2017. A comprehensive survey on query expansion techniques, their issues and challenges. *International Journal of Computer Applications*, 168(12).

Douwe Kiela, Felix Hill, and Stephen Clark. 2015. Specializing word embeddings for similarity or relatedness. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2044–2048.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(4):359–389.

Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. 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 (NAACL-HLT)*, pages 970–976.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to wordnet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–244.

Eric Nalisnick, Bhaskar Mitra, Nick Craswell, and Rich Caruana. 2016. Improving document ranking with dual word embeddings. In *25th International Conference on World Wide Web (WWW)*, pages 83–84.

Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017a. Hierarchical embeddings for hypernymy detection and directionality. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 233–243.

Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017b. Distinguishing antonyms and synonyms in a pattern-based neural network. In *15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 76–85.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Conference on Empirical Methods on Natural Language Processing (EMNLP)*, pages 1532–1543.

Stephen Roller, Katrin Erk, and Gemma Boleda. 2014. Inclusive yet selective: Supervised distributional hypernymy detection. In *25th International Conference on Computational Linguistics (COLING)*, pages 1025–1036.

Sebastian Ruder and Barbara Plank. 2018. Strong baselines for neural semi-supervised learning under domain shift. In *56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Enrico Santus, Alessandro Lenci, Tin-Shing Chiu, Qin Lu, and Chu-Ren Huang. 2016. Nine features in a random forest to learn taxonomical semantic relations. In *10th International Conference on Language Resources and Evaluation (LREC)*, pages 4557–4564.

Enrico Santus, Vered Shwartz, and Dominik Schlechtweg. 2017. Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. In *15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 65–75.

Enrico Santus, Frances Yung, Alessandro Lenci, and Chu-Ren Huang. 2015. 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)*, pages 64–69.

Vered Shwartz and Ido Dagan. 2016. Cogalex-v shared task: Lexnet - integrated path-based and distributional method for the identification of semantic relations. *CoRR*, abs/1610.08694.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2389–2398.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. In *17th International Conference on Neural Information Processing Systems (NIPS)*, pages 1297–1304.

Robert Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *31st Conference on Artificial Intelligence (AAAI)*, pages 4444–4451.

Ekaterina Vylomova, Laura Rimell, Trevor Cohn, and Timothy Baldwin. 2016. 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 (ACL)*, pages 1671–1682.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David J. Weir, and Bill Keller. 2014. Learning to distinguish hypernyms and co-hyponyms. In *5th International Conference on Computational Linguistics (COLING)*, pages 2249–2259.

Zheng Yu, Haixun Wang, Xuemin Lin, and Min Wang. 2015. Learning term embeddings for hypernymy identification. In *24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1390–1397.