Patch-Based Identification of Lexical Semantic Relations

Nesrine Bannour[1], Gaël Dias[1(✉)], Youssef Chahir[1], and Houssam Akhmouch[1,2]

[1] Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France
gael.dias@unicaen.fr
[2] Crédit Agricole Brie Picardie, 77700 Serris, France

Abstract. The identification of lexical semantic relations is of the utmost importance to enhance reasoning capacities of Natural Language Processing and Information Retrieval systems. Within this context, successful results have been achieved based on the distributional hypothesis and/or the paradigmatic assumption. However, both strategies solely rely on the input words to predict the lexical semantic relation. In this paper, we make the hypothesis that the decision process should not only rely on the input words but also on their K closest neighbors in some semantic space. For that purpose, we present different binary and multi-task classification strategies that include two distinct attention mechanisms based on PageRank. Evaluation results over four gold-standard datasets show that average improvements of 10.6% for binary and 8% for multi-task classification can be achieved over baseline approaches in terms of F_1. The code and the datasets are available upon demand.

Keywords: Patches · PageRank · Attention mechanism · Multi-task learning

1 Introduction

Recognizing the exact nature of the semantic relation holding between a pair of words is crucial for many applications such as taxonomy induction [10], question answering [6,22], query expansion [15] or text summarization [8]. The most studied lexical semantic relations are synonymy, co-hyponymy, hypernymy, or meronymy, but more exist [37]. Numerous approaches have been proposed to identify one particular semantic relation of interest following either the paradigmatic approach [28,29,33,39], the distributional model [9,31,36,37], or their combination [25,32].

In all these studies, the decision process relies on finding representation regularities between two input words. In this paper, we make the assumption that finding the lexical semantic relation that holds between two words does not solely rely on the pair itself, but also on the semantically related neighboring words. Our hypothesis relies on two different ideas. First, studies about the mental lexicon [13,23] theorize that words are highly interconnected within a mental semantic network, such that conceptual information is encoded in one's mind rather than single words alone. Second, studies in image segmentation show that dividing up an image into a patch work of regions, each of which being homogeneous, leads to successful results [18,20]. Analogously, we propose to define a word patch as a source word augmented by its semantically close related neighbors, and expect that performance gains can be achieved by grounding the decision process on finding representation regularities between word patches.

However, as the information contained in patches may be voluminous, noisy information may be embedded possibly due to semantic shifts, so that concept centrality may be lost. Consequently, we propose to define two attention mechanisms (one inside patches and one between patches) based on the PageRank algorithm [26] to account for the valuable information present in large patch-based input representation vectors.

In order to test our hypotheses, we follow the distributional approach, although we acknowledge that its combination with the pattern-based approach could lead to improved results as stated in [25,32]. However, integrating continuous pattern representations within the patch paradigm is not straightforward and requires specific further analysis, which is out of the scope of this paper. But, to overcome the main drawback of the distributional hypothesis that conflates different semantic relations between words [9], we design different multi-task neural learning strategies, as recently introduced by [1], together with their binary classification counterparts.

Results over four gold-standard datasets i.e. RUMEN [1], ROOT9 [30], WEEDS [39] and BLESS [3] show that the patch representation leads to significant improvements, in particular when attention mechanisms are applied - 10.6% for binary and 8% for multi-task classification over baselines in terms of F_1 score. Moreover, multi-task learning strategies evidence slightly improved performance results as well as more coherent behaviors when compared to binary configurations.

2 Related Work

Two main approaches have been intensively studied to classify word pairs into the lexical semantic relation they share, or to categorize them as unrelated (or random). On the one hand, pattern-based methods rely on lexico-syntactic patterns, which connect a pair of words [12,17,25,29,32,33]. On the other hand, the distributional approach consists in characterizing the semantic relation between two words based on their distributional representations, thus following Harris distributional hypothesis [11]. In this case, a word pair can be represented by the concatenation of the context vectors of the individual words [2,28,32,39] or by their difference [7,37,39]. The main drawback of the distributional hypothesis is that it conflates different semantic relations between words. Therefore, different solutions have been proposed to overcome this issue. First, specialized similarity measures can be defined to distinguish different relations [29]. Another solution is to specialize word embeddings for particular relations using external knowledge [9,36]. However, these methods are one-relation specific and cannot differentiate between multiple semantic relations at a time. So, multi-task learning strategies have been proposed [1], which concurrently learn different semantic relations with the assumption that the learning process of a given semantic relation may be improved by jointly learning another semantic relation.

In this paper, we propose to look at the problem from a different point of view, the underlying idea being that if each word is augmented by its set of K nearest neighbors (i.e. a patch), improved performance results may be attained. A similar idea is proposed by [14], who presented a set cardinality-based method, which exploits WordNet [24] to compute related neighboring words. In particular, they show that the features extracted from set cardinalities produce competitive results compared to word embedding approaches for a large set of word similarity tasks. However, their methodology relies on the pre-existence of a knowledge base, which is not available for a vast majority of languages. Moreover, their hypothesis builds on discrete representations of words, which cannot account for word continuous similarities and thus highly relies on representative[1] set intersections. Also, only word similarity tasks are tested and it is difficult to access to what extent their methodology can adapt to lexical semantic relation identification. Furthermore, in their proposal, all neighboring words receive the same importance for the decision process, although by extending the semantic scope of each individual word, semantic shifts may occur as well as concept centrality may be lost.

To overcome all these situations, we propose binary and multi-task classification strategies grounded on continuous input representations, that combine two attention mechanisms to evidence word centrality within and between patches based on the PageRank algorithm. In particular, word pairs are represented by the concatenation of their word embeddings as suggested by [32], augmented by their cosine similarity, which is an important feature for lexical semantic relation identification [31].

3 Patch-Based Classification

In this section, we present the overall learning architecture, which consists in the definition of (1) the new input representations based on patches, (2) two attention mechanisms grounded on the PageRank algorithm, and (3) the binary and multi-task neural parallel and sequential classification configurations.

3.1 Definition of Patch and Similarity Between Patches

Definition of Patch. A patch consists of the K most similar words w_j to a source word w_0 in terms of cosine similarity in some latent semantic space (embedding). As such, the patch $P^K_{w_0}$ corresponding to the source word w_0 is a set of $K+1$ words defined as in Eq. 1, where $cos(\overrightarrow{w_0}, \overrightarrow{w_j})$ stands for the cosine similarity between the two representation vectors of w_0 and w_j in some semantic space.

$$P^K_{w_0} = \{w_0\} \cup \{w_j | argmax^K_{cos(\overrightarrow{w_0}, \overrightarrow{w_j})}\} \tag{1}$$

[1] In terms of occurrence and variety.

The next step consists in transforming a patch into a learning input. For that purpose, we follow a simple strategy that consists in concatenating the distributional representations of all words within a patch in their order of similarity measure with the source word[2]. Such input is defined in Eq. 2, where $\overrightarrow{w_i}$ stands for the distributional representation of the i_{th} word in the patch $P^K_{w_0}$.

$$\bigoplus_{i=0}^{K} \overrightarrow{w_i} \tag{2}$$

Similarity Between Patches. Many similarity measures exist to account for the lexical semantic relation that links two words [31]. Within this scope, the cosine similarity measure has evidenced successful results for a variety of semantic relations [35]. As a consequence, we propose to extend the cosine similarity to patches in a straightforward manner. The similarity between two patches is the set of one-to-one cosine similarity measures between all words in their respective patches. It is formally defined in Eq. 3, where $P^K_{w_0}$ and $P^K_{w'_0}$ are two patches, and the $(K+1) \times (K+1)$ matrix noted $SP(P^K_{w_0}, P^K_{w'_0})$ summarizes all values.

$$SP(P^K_{w_0}, P^K_{w'_0}) = \begin{bmatrix} cos(\overrightarrow{w_0}, \overrightarrow{w'_0}) & cos(\overrightarrow{w_0}, \overrightarrow{w'_1}) & \dots & cos(\overrightarrow{w_0}, \overrightarrow{w'_K}) \\ cos(\overrightarrow{w_1}, \overrightarrow{w'_0}) & cos(\overrightarrow{w_1}, \overrightarrow{w'_1}) & \dots & cos(\overrightarrow{w_1}, \overrightarrow{w'_K}) \\ \vdots & \vdots & \ddots & \vdots \\ cos(\overrightarrow{w_K}, \overrightarrow{w'_0}) & cos(\overrightarrow{w_K}, \overrightarrow{w'_1}) & \dots & cos(\overrightarrow{w_K}, \overrightarrow{w'_K}) \end{bmatrix} \tag{3}$$

Similarly to the transformation of a patch into a learning input, we concatenate all $(K+1) \times (K+1)$ values of the $SP(P^K_{w_0}, P^K_{w'_0})$ matrix to be fed to the decision process. Such input is defined in Eq. 4.

$$\bigoplus_{i=0}^{K} \bigoplus_{j=0}^{K} cos(\overrightarrow{w_i}, \overrightarrow{w'_j}) \tag{4}$$

3.2 Attention Mechanisms

Attention Mechanism Within a Patch. A patch should ideally represent a semantic concept centered around its source word. However, this may not be the case as semantic shifts may occur when augmenting the source word with K likely related neighbors, such that centrality may be lost. In order to measure centrality within a patch, we propose to run the PageRank algorithm [26] over the patch defined as a weighted complete graph. Thus, a patch is defined as $G_{P^K_{w_0}} = (V_{P^K_{w_0}}, E_{P^K_{w_0}})$, where $V_{P^K_{w_0}}$ is the set of $K+1$ vertices (words) within $P^K_{w_0}$, and $E_{P^K_{w_0}}$ is the complete set of edges that link all vertices together, weighted by their corresponding cosine similarity. The result of the PageRank algorithm over

[2] Other representations have been tested, but this solution proved to lead to better results.

$G_{P_{w_0}^K}$ is a vector of $(K+1)$ dimensions, where each vertex (word) within the patch receives a centrality score in $\mathbb{R}$, and it is noted $\overrightarrow{\alpha_{w_0}^K} = \langle \alpha_{w_0}, \alpha_{w_1}, \alpha_{w_2}, \ldots, \alpha_{w_K} \rangle$.

The output of the PageRank algorithm $\overrightarrow{\alpha_{w_0}^K}$ stands for the attention mechanism within a patch. Indeed, if we represent a patch (as a learning input) by the concatenation of its words embeddings, all input words are given the same importance for the decision process, letting the classification algorithm decide upon which input dimensions should be discriminant. To guide the learning process, attention mechanisms have shown successful results [34]. As a consequence, we propose to weight each input embedding by its PageRank value so that word centrality scores are included in the decision process. Such an attention-based input is defined in Eq. 5.

$$\bigoplus_{i=0}^{K} \alpha_{w_i}.\overrightarrow{w_i} \tag{5}$$

Attention Mechanism Between Patches. While the first attention mechanism focuses on word centrality within a given patch, the second attention mechanism spotlights on word centrality between patches. Indeed, it is important to acknowledge, which words are central in a set of two patches in order to verify if two words are in a lexical semantic relation. As such, if two patches share a set of close semantically related words that are central to both concepts, the decision process may be more reliable. In order to measure word centrality between two patches ($P_{w_0}^K$, $P_{w_0'}^K$), we propose to run the PageRank algorithm over the graph defined by the $SP(P_{w_0}^K, P_{w_0'}^K)$ matrix, which results in a vector of $2 \times (K+1)$ dimensions, where each word of both patches receives a centrality score in $\mathbb{R}$, and it is noted $\overrightarrow{\beta_{w_0,w_0'}^K} = \langle \beta_{w_0}, \beta_{w_1}, \ldots, \beta_{w_K}, \beta_{w_0'}, \beta_{w_1'}, \ldots, \beta_{w_K'} \rangle$.
The output of the PageRank algorithm $\overrightarrow{\beta_{w_0,w_0'}^K}$ stands for the attention mechanism between patches. So, similarly to the previous attention mechanism, we propose to weight each input embedding of a pair of patches by its PageRank value so that inter-patch word centrality scores are fed to the decision process. Such a second attention-based input is defined in Eq. 6.

$$\bigoplus_{i=0}^{K} \beta_{w_i}.\overrightarrow{w_i} \bigoplus_{i=0}^{K} \beta_{w_i'}.\overrightarrow{w_i'} \tag{6}$$

Combined Attention Mechanisms. Based on the previous definitions of attention mechanisms, we can acknowledge that all input word embeddings may receive two centrality scores: one within patches (first attention) and one between patches (second attention). As a consequence, both attention mechanisms can be combined in a unique learning representation to be fed to the decision process, and it is defined in Eq. 7.

$$\bigoplus_{i=0}^{K} \alpha_{w_i}.\beta_{w_i}.\overrightarrow{w_i} \bigoplus_{i=0}^{K} \alpha_{w'_i}.\beta_{w'_i}.\overrightarrow{w'_i} \tag{7}$$

3.3 Learning Framework

In order to perform binary and multi-task classification, we define two distinct learning input representations, X_p and X_s, which combine patches representations, attention mechanisms and similarity between patches. The first learning input configuration X_p builds on the individual attention mechanisms and it is defined in Eq. 8. In particular, it is composed of the concatenated embeddings of $P_{w_0}^K$ and $P_{w'_0}^K$ weighted by their inside patch attention, plus the concatenated embeddings of $P_{w_0}^K$ and $P_{w'_0}^K$ weighted by their in between patch attention, plus the concatenated values of the cosine similarity between both patches $P_{w_0}^K$ and $P_{w'_0}^K$.

$$X_p = (\bigoplus_{i=0}^{K} \alpha_{w_i}.\overrightarrow{w_i}, \bigoplus_{j=0}^{K} \alpha_{w'_j}.\overrightarrow{w'_j}, \bigoplus_{i=0}^{K} \beta_{w_i}.\overrightarrow{w_i} \bigoplus_{i=0}^{K} \beta_{w'_i}.\overrightarrow{w'_i}, \bigoplus_{i=0}^{K}\bigoplus_{j=0}^{K} cos(\overrightarrow{w_i}, \overrightarrow{w'_j})) \tag{8}$$

The second learning input X_s is grounded on the combined attention mechanism, which allows a more compact representation. It is defined in Eq. 9 and it consists in the concatenated embeddings of $P_{w_0}^K$ and $P_{w'_0}^K$ weighted by their combined inside and in between patch attentions, plus the concatenated values of the cosine similarity between both patches $P_{w_0}^K$ and $P_{w'_0}^K$.

$$X_s = (\bigoplus_{i=0}^{K} \alpha_{w_i}.\beta_{w_i}.\overrightarrow{w_i} \bigoplus_{i=0}^{K} \alpha_{w'_i}.\beta_{w'_i}.\overrightarrow{w'_i}, \bigoplus_{i=0}^{K}\bigoplus_{j=0}^{K} cos(\overrightarrow{w_i}, \overrightarrow{w'_j})) \tag{9}$$

With respect to binary and multi-task classification algorithms, we adapted the (hard parameter sharing architecture) feed-forward neural networks proposed in [1], as the code is freely available[3], and as a consequence allows direct comparison and reproducibility. In particular, Adam [16] is used as the optimizer with default parameters of Keras [5]. For multi-task classification, the network is trained with batches of 64 examples and the number of iterations is optimized to maximize the F_1 score on the validation set. Word embeddings are initialized with the 300-dimensional representations of GloVe [27]. The overall architectures are illustrated in Fig. 1 both for X_p (parallel architecture) and X_s (sequential architecture).

4 Evaluation Results

In this section, we present overall classification results for four gold-standard datasets, namely RUMEN [1], ROOT9 [32], WEEDS [39] and BLESS [3].

[3] https://bit.ly/2Qitasd.

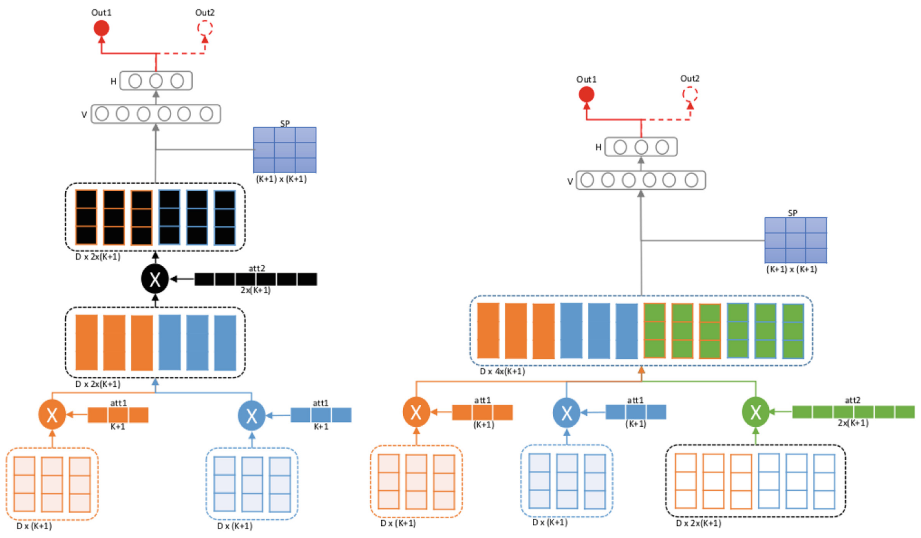

Fig. 1. On the **left**, the sequential architecture. On the **right**, the parallel architecture. Note that the dotted red line between the last fully connected layer of the neural network and the output layer stands for the multi-task architecture only. (Color figure online)

In particular, we test 8 different configurations for both binary and multi-task classification strategies: (1) *Concat*, where the learning input is the concatenation of the word pair embeddings, (2) *Concat + cos* stands for the same configuration as *Concat* plus the cosine similarity measure between the two word embeddings, (3) *Patches* consists of the concatenation of the embeddings of the word pair plus all its respective neighbors, (4) *Concat + SP* is similar to *Concat + cos*, but the cosine similarity is replaced by the concatenated *SP* matrix, (5) *Patches + SP* represents the exact counterpart of the *Concat + cos* input for patches, and combines the concatenation of all embeddings within patches plus the concatenated *SP* matrix, (6) *Patches + SP +* att_1 is similar to *Patches + SP*, where each individual patch is weighted by the attention mechanism within patch, (7) *Patches + SP +* att_{12} represents the sequential architecture illustrated in Fig. 1 (left), and (8) *Patches + SP +* att_{1+2} stands for the parallel architecture illustrated in Fig. 1 (right). Classification performance is evaluated through Accuracy, F_1 score, Precision and Recall. Note that lexical split is applied to avoid vocabulary intersection between training, validation and test datasets to avoid lexical memorization [19].

Binary Classification: Results for binary classification are given in Table 1. They clearly evidence the superiority of the sequential architecture, which produces best results in 6 learning situations out of 8 in terms of F_1 score[4]. In particular,

[4] F_1 is not sensitive to unbalanced datasets.

Table 1. Accuracy, F_1, Precision and Recall scores on all datasets for binary classification.

			Synonym vs Random					Hypernym vs Random			
	Inputs		Acc	F_1	Prec	Rec		Acc	F_1	Prec	Rec
RUMEN	Concat	–	0.756	0.760	0.744	0.776	–	0.776	0.768	0.741	0.714
	Concat + cos	–	0.817	0.819	0.806	0.833	–	0.790	0.766	0.741	0.792
	Patches	K = 1	0.713	0.718	0.701	0.736	K = 3	0.765	0.728	0.728	0.729
	Concat + SP	K = 6	0.829	0.830	0.819	**0.841**	K = 9	0.801	0.778	0.753	**0.805**
	Patches + SP	K = 1	0.792	0.793	0.786	0.799	K = 7	0.779	0.747	0.740	0.754
	Patches + SP + att_1	K = 9	0.833	0.831	0.835	0.828	K = 2	0.818	**0.789**	0.792	0.786
	Patches + SP + att_{1+2}	K = 9	0.830	0.827	0.839	0.814	K = 2	0.811	0.780	0.782	0.779
	Patches + SP + att_{12}	K = 9	**0.855**	**0.851**	**0.873**	0.829	K = 4	**0.819**	0.783	**0.813**	0.756
			Co-hynonym vs Random					Hypernym vs Random			
	Inputs		Acc	F_1	Prec	Rec		Acc	F_1	Prec	Rec.
ROOT9	Concat	–	0.907	0.938	0.949	0.888	–	0.852	0.895	0.901	0.927
	Concat + cos	–	0.911	0.941	0.951	0.931	–	0.835	0.883	0.891	0.874
	Patches	K = 10	0.911	0.941	0.948	0.935	K = 2	0.876	0.913	0.908	0.918
	Concat + SP	K = 10	0.926	0.951	0.955	0.947	K = 9	0.893	0.924	0.927	0.942
	Patches + SP	K = 10	0.920	0.946	0.957	0.936	K = 6	0.903	0.931	0.938	0.923
	Patches + SP + att_1	K = 9	0.945	0.964	0.961	0.966	K = 3	0.919	0.943	0.938	0.949
	Patches + SP + att_{1+2}	K = 9	**0.950**	**0.967**	0.965	**0.970**	K = 6	0.908	0.934	**0.943**	0.925
	Patches + SP + att_{12}	K = 3	0.946	0.964	**0.968**	0.961	K = 2	**0.931**	**0.952**	0.941	**0.963**
			Co-hyponym vs Random					Hypernym vs Random			
	Inputs		Acc	F_1	Prec	Rec		Acc	F_1	Prec	Rec.
WEEDS	Concat	–	0.718	0.429	0.414	0.444	–	0.810	0.422	0.433	0.412
	Concat + cos	–	0.789	0.587	0.548	0.632	–	0.842	0.527	0.530	0.524
	Patches	K = 8	0.711	0.406	0.397	0.416	K = 1	0.789	0.376	0.374	0.377
	Concat + SP	K = 9	0.841	0.671	0.660	**0.682**	K = 9	0.901	0.653	0.796	0.554
	Patches + SP	K = 9	0.794	0.592	0.560	0,628	K = 9	0.871	0.602	0.623	0.583
	Patches + SP + att_1	K = 8	0.859	0.680	**0.789**	0.629	K = 7	0.904	0.672	0.793	0.583
	Patches + SP + att_{1+2}	K = 10	0.861	0.685	0.743	0.635	K = 9	0.903	0.661	0.799	0.564
	Patches + SP + att_{12}	K = 4	**0.870**	**0.713**	0.751	0.679	K = 9	**0.926**	**0.753**	**0.856**	**0.670**
			Meronym vs Random					Hypernym vs Random			
	Inputs		Acc	F_1	Prec	Rec		Acc	F_1	Prec	Rec.
BLESS	Concat	–	0.844	0.757	0.804	0.716	–	0.868	0.468	0.515	0.429
	Concat + cos	–	0.862	0.785	0.833	0.743	–	0.881	0.521	0.568	0.481
	Patches	K = 3	0.848	0.764	0.808	0.725	K = 2	0.922	0.692	0.742	0.647
	Concat + SP	K = 8	0.887	0.829	0.857	0.803	K = 4	0.933	0.741	0.780	0.705
	Patches + SP	K = 7	0.881	0.818	0.855	0.785	K = 2	0.939	0.759	0.806	0.718
	Patches + SP + att_1	K = 9	0.896	0.835	0.905	0.776	K = 3	0.949	0.794	0.870	0.731
	Patches + SP + att_{1+2}	K = 3	0.889	0.835	0.850	**0.820**	K = 2	0.948	0.798	0.838	0.763
	Patches + SP + att_{12}	K = 6	**0.898**	**0.837**	**0.915**	0.772	K = 2	**0.955**	**0.822**	**0.882**	**0.769**

for RUMEN, it outperforms the second best strategy by 2% for synonymy. For ROOT9, improvements of 0.9% are obtained for hypernymy. For WEEDS, increases of 2.8% for co-hyponymy and 8.1% for hypernymy are obtained. For BLESS, enhancements of 0.2% and 2.4% are respectively achieved for meronymy and hypernymy over the second best approach. Interestingly, the parallel architecture is not capable of taking advantage of the second attention mechanism (i.e. centrality between patches) as it is the case for the sequential model. The inability of the parallel architecture to combine both attention mechanisms can be explained by two factors. First, in the parallel architecture, individual word embeddings can receive different PageRank weights depending on the attention mechanism. But, as these weights are not explicitly combined, the neural network may not be able to compute regularities between different weights for the same input. The other reason can be explained by the tree structure of human knowledge [38], that theorizes that the cognitive process of human acquisition is agglomerative. Results also evidence clear superiority of attention-based strategies compared to attention-unaware patch-based models i.e., *Patches, Concat + SP* and *Patches + SP*. On average overall all datasets and all classification tasks, an increase of 3.5% is achieved by the best attention-based model compared to the best attention-unaware configuration containing neighbor information. Moreover, if we compare the sequential architecture to the current non-patch baseline i.e. *Concat + cos*, the difference in performance is much more important. In particular, an average improvement of 10.6% can be attained, with a minimum increase of 1.7% for RUMEN (hypernymy) and a maximum gain of 30.1% for BLESS (hypernymy). Results also show that the simple introduction of neighbors, i.e. without attention mechanisms, cannot account systematically for increase in performance. Indeed, if we compare *Concat + cos* and its direct counterpart *Patches + SP* that includes the neighbor information both in terms of semantic content and similarity, best results are obtained for the second strategy in 6 cases out of 8. This means that in two situations, the introduction of more information does not lead to improvements. By looking in more details and comparing *Concat* to *Patches*, we can see that better results can be obtained in only half of the cases by the introduction of neighbor embeddings. In fact, going further in the analysis and looking at the results of *Concat + SP*, we clearly understand that the improvement in results comes from the SP matrix and not from the concatenation of the embeddings. Indeed, the *Concat + SP* strategy shows improved results in 6 cases out of 8 over the more complete *Patches + SP* configuration, with an average increase of 3.6%. This situation can be explained by the inability of the neural network to focus on the meaningful word embeddings. Indeed, by just concatenating all neighbor embeddings, all words become equal for the decision process, although this should not be the case. As such, attention mechanisms allow to overcome this situation.

Finally, we can observe that different values of K are obtained for all tested situations. First, if we compare the best attention-based model with the best attention-unaware configuration[5] for all the classification tasks, it is clear that smaller values of K are needed for models with attention. On average, attention-aware models find optimal results for $K = 5.4$, while configurations without attention attain maximum performance for $K = 7.4$. This situation can easily be explained as the best attention-unaware model is *Concat + SP*, which does not include the embeddings of the neighbors. As such, the only information from the neighbors is given by the SP matrix, which must give trace of all the information alone between patches. As a consequence, only large matrices can account for the extra information included in the patches. Results also show that different values of K can be obtained for the same semantic

[5] Which may not include any information about neighbors.

relation depending on the dataset, although some regularities seem to exist. For instance, for hypernymy, best results are obtained for $K = 9$ for WEEDS, but the best results for RUMEN, BLESS and ROOT9 are obtained for $K = 2$. Furthermore, by comparing symmetric (synonymy, co-hyponymy) and asymmetric (hypernymy, meronymy) relations, results show that best results are obtained for $K = 7.4$ on average in the first case, and for $K = 6$ on average in the second case, thus suggesting the need for less extra-knowledge for asymmetric semantic relations. This can be explained by the semantic shift phenomenon [4]. When dealing with asymmetric relations, it is likely that one of the words within the pair is a general word. As a consequence, when expanding the general word with its neighbors, it is likely that a semantic shift occurs. As a consequence, it is conceivable that smaller values of K are preferred for asymmetric relations.

Multi-task Classification: Results for multi-task classification are given in Table 2. They clearly evidence the increase in performance from the sequential architecture compared to all other strategies. Indeed, best results are achieved by this configuration in almost all cases. In particular, average improvements of 15.4% and 8.1% are attained when compared to the recent work of [1] i.e. *FS Concat* and its extended version including the cosine similarity measure, i.e. *FS Concat + cos*. Results also show that similar behaviors to binary classification can be observed. First, the introduction of the cosine similarity measure greatly boosts performance. Second, most of the information gain obtained by strategies without attention mechanisms comes from the *SP* matrix and not from the concatenation of the embeddings of the neighbors present in a given patch. Note that in this case, that *Concat + SP* is the best attention-unaware strategy, which achieves better results in terms of F_1 than *Patches + SP +* att_1 in 2 cases out of 8. Third, if we compare the sequential architecture to the current multi-task non-patch baseline, i.e. *FS Concat + cos*, similar differences in performance are obtained compared to the binary situation. In particular, an average improvement of 8% can be attained, with a minimum increase of 1.7% for RUMEN (hypernymy) and a maximum gain of 21.1% for BLESS (hypernymy). Finally, likewise the binary situation, the parallel architecture can not compete with the sequential counterpart and does not outperform the configuration with only within patch attention, i.e. *Patches + SP +* att_1. When compared to the sequential binary classification model, the multi-task sequential counterpart based on the fully-shared architecture evidences best results in 5 cases out of 8. Nevertheless, overall improvements are small with an average increase in performance of 0.7%. Similarly, in the 3 other cases where the binary strategy offers best results, average gains of 0.4% are obtained. This can easily be explained by the small difference in architectures as already evidenced in [1]. Interestingly, the multi-task model evidences steady improvements in terms of Recall, with best values than the binary counterpart in 7 cases out of 8. In parallel, Precision shows worst results in 6 cases out of 8. This can also be explained by the fully-shared architecture that produces its decision based on a single shared layer, i.e. features that could be specific to one single task maybe lost. As a consequence, Precision may be affected while Recall boosted.

Table 2. Accuracy, F_1, Precision and Recall scores on all datasets for multi-task classification.

			Synonym vs Random				Hypernym vs Random			
	Inputs	K	Acc	F_1	Prec	Rec	Acc	F_1	Prec	Rec
RUMEN	FS Concat [1]	–	0.740↓	0.744↓	0.727↓	0.764↓	0.790↑	0.763↓	0.746↑	0.780↑
	FS Concat [1] + cos	–	0.824↑	0.827↑	0.810↑	0.844↑	0.794↑	0.769↑	0.748↑	0.792=
	FS Patches	1	0.717↑	0.722↑	0.704↑	0.742↑	0.742↓	0.705↓	0.696↓	0.715↓
	FS Concat + SP	9	0.835↑	0.834↑	0.834↑	0.835↓	0.807↑	0.780↑	0.767↑	0.792↓
	FS Patches + SP	1	0.800↑	0.801↑	0.790↑	0.812↑	0.775↓	0.743↓	0.733↓	0.753↓
	FS Patches + SP + att_1	4	0.828↓	0.823↓	0.840↑	0.807↓	0.815↓	0.782↓	0.795↑	0.769↓
	FS Patches + SP + att_{1+2}	4	0.820↓	0.816↓	0.829↓	0.802↓	0.815↑	**0.790**↑	0.775↓	**0.806** ↑
	FS Patches + SP + att_{12}	8	**0.852**↓	**0.850**↓	**0.861**↓	**0.839**↑	**0.821**↑	0.786↑	**0.810**↓	0.764↑
			Co-hyponym vs Random				Hypernym vs Random			
	Inputs		Acc	F_1	Prec	Rec	Acc	F_1	Prec	Rec.
ROOT9	FS Concat [1]	–	0.896↓	0.932↓	0.936↓	0.928↑	0.807↓	0.863↓	0.868↓	0.858↓
	FS Concat [1] + cos	–	0.901↓	0.935↓	0.938↓	0.933↑	0.835=	0.882↓	0.893↑	0.872↓
	FS Patches	1	0.903↓	0.937↓	0.932↓	0.942↑	0.850↓	0.895↓	0.888↓	0.902↓
	FS Concat + SP	5	0.922↓	0.949↓	0.952↓	0.945↓	0.883↓	0.917↓	0.918↓	0.916↓
	FS Patches + SP	2	0.917↓	0.946=	0.941↓	0.951↓	0.898↓	0.929↓	0.918↓	0.940↑
	FS Patches + SP + att_1	3	0.926↓	0.951↓	0.954↑	0.949↓	0.909↓	0.937↓	0.925↓	0.949=
	FS Patches + SP + att_{1+2}	5	0.919↓	0.947↓	0.944↓	0.950↓	0.914↑	0.939↑	**0.941**↓	0.937↑
	FS Patches + SP + att_{12}	3	**0.941**↓	**0.961**↓	**0.960**↓	**0.963**↑	**0.921**↓	**0.945**↓	0.938↓	**0.951**↓
			Co-hyponym vs Random				Hypernym vs Random			
	Inputs		Acc	F_1	Prec	Rec	Acc	F_1	Prec	Rec.
WEEDS	FS Concat [1]	–	0.732↑	0.446↑	0.439↑	0.454↑	0.829↑	0.455↑	0.489↑	0.426↑
	FS Concat [1] + cos	–	0.810↑	0.627↑	0.615↑	0.670↑	0.871↑	0.616↑	0.615↑	0.618↑
	FS Patches	1	0.701↓	0.391↓	0.379↓	0.403↓	0.816↑	0.415↑	0.446↑	0.387↑
	FS Concat + SP	8	0.841=	0.672↑	0.659↓	0.686↑	0.897↓	0.668↑	0.728↓	0.618↑
	FS Patches + SP	9	0.800↑	0.604↑	0.571↑	0.641↑	0.827↓	0.563↓	0.489↓	0.662↑
	FS Patches + SP + att_1	3	0.832↓	0.657↓	0.640↓	0.676↑	0.907↑	0.685↑	0.794↑	0.603↑
	FS Patches + SP + att_{1+2}	5	0.853↓	0.679↓	0.709↓	0.670↑	0.877↓	0.639↓	0.631↓	0.647↑
	FS Patches + SP + att_{12}	6	**0.876**↑	**0.731**↑	**0.756**↑	**0.708**↑	**0.927**↑	**0.759**↑	**0.848**↓	**0.686**↑
			Meronym vs Random				Hypernym vs Random			
	Inputs		Acc	F_1	Prec	Rec	Acc	F_1	Prec	Rec.
BLESS	FS Concat [1]	–	0.835↓	0.740↓	0.800↓	0.689↓	0.891↑	0.526↑	0.636↑	0.448↑
	FS Concat [1] + cos	–	0.862=	0.783↓	0.845↑	0.729↓	0.908↑	0.616↑	0.699↑	0.551↑
	FS Patches	2	0.842↓	0.750↓	0.817↑	0.692↓	0.927↑	0.695↑	0.789↑	0.622↓
	FS Concat + SP	8	0.882↓	0.816↓	0.872↑	0.766↓	0.931↓	0.722↓	0.788↑	0.667↓
	FS Patches + SP	2	0.853↓	0.769↓	0.828↓	0.718↓	0.942↑	0.774↑	0.816↑	0.737↑
	FS Patches + SP + att_1	3	0.872↓	0.803↓	0.844↓	0.766↑	0.955↑	0.818↑	0.900↑	**0.750**↑
	FS Patches + SP + att_{1+2}	3	0.867↓	0.794↓	0.843↓	0.750↓	0.951↑	0.801↑	0.878↑	0.737↓
	FS Patches + SP + att_{12}	3	**0.895**↓	**0.839**↑	**0.882**↓	**0.789**↑	**0.958**↑	**0.827**↑	**0.921**↑	**0.750**↓

In order to better understand the impact of K on the results, we show performance results for the binary and multi-task sequential architectures for $K = 1..10$ in Fig. 2. Note that results in Table 2 are given for the best K on average and cannot account for differences in K between two semantic relations. Overall, we can notice different behaviors depending on the dataset. For RUMEN, performance values vary minimally with respect to K, independently of the semantic relation. For ROOT9 and BLESS, similar situations can be observed. Indeed, for both datasets, performance highly degrades with higher values of K for hypernymy,

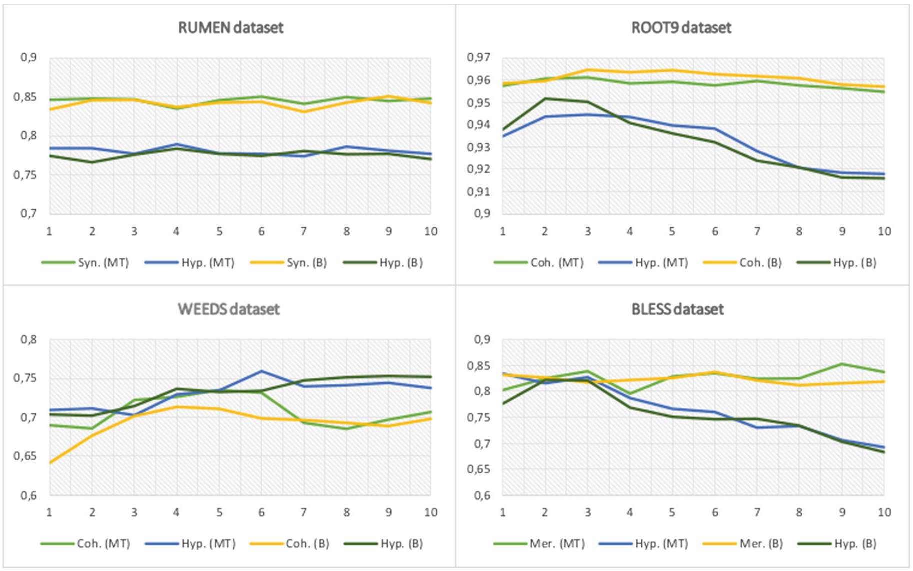

Fig. 2. F_1 score distribution by K ($K = 1..10$) for all datasets for the sequential architecture, both for binary (B) and multi-task (MT) classification.

while the impact of K on co-hyponymy is more limited. For WEEDS, the situation is the opposite for hypernymy and co-hyponymy, as higher performance values are obtained for higher values of K. As a consequence, no clear conclusion can be drawn as results drastically change depending on the dataset. It is clear that further efforts are needed to propose better evaluation standards. Nevertheless, some regularities emerge. Indeed, over all values of K, the multi-task architecture evidences steady improvements over the binary counterpart for all datasets, except for ROOT9 where best results are obtained by the binary classification for both semantic relations. Also, in 2 cases out of 8, the best value of K is the same for the binary and the multi-task situations. In 3 cases, the difference in K value is one, and in the remaining 3 cases, the difference of K equals to 3, with 2 cases in which the multi-task architecture needs less neighbors than the binary version. This confirms the fact that small differences in terms of K values are evidenced on average between both strategies, although there seems to be a slight tendency to use less neighbors in the multi-task strategy.

5 Conclusions

In this paper, we presented a patch-based classification strategy to tackle lexical semantic relation identification. In particular, we showed that attention mechanisms (if correctly combined) drastically boost results compared to attention-unaware configurations. Indeed, average improvements can reach 10.6% for binary and 8% for multi-task classification over non-patch baseline approaches in terms of F_1 for the sequential architecture, when tested over four gold-standard datasets, namely RUMEN, ROOT9, WEEDS and BLESS. Moreover, results witness that small but steady improvements in classification performance can be attained by multi-task architectures. As such, immediate future work should include (1) the design of new multi-task architectures following the ideas of [21], (2) the combination of the distributional approach with the paradigmatic model as suggested in [25], and (3) the evaluation comparison to the very recent work proposed by [14], which follows similar ideas with discrete word representations.

References

1. Balikas, G., Dias, G., Moraliyski, R., Akhmouch, H., Amini, M.-R.: Learning lexical-semantic relations using intuitive cognitive links. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11437, pp. 3–18. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15712-8_1
2. Baroni, M., Bernardi, R., Do, N.Q., Shan, C.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)
3. 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)
4. Blank, A.: Why do new meanings occur? A cognitive typology of the motivations for lexical semantic change. Cogn. Linguist. Res. **13**, 61–90 (1999)
5. Chollet, F.: Keras. https://keras.io (2015)
6. 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. 875–886 (2017)
7. Fu, R., Guo, J., Zhao, Y., Che, W., Wang, H., Liu, T.: Learning semantic hierarchies: a continuous vector space approach. IEEE/ACM Trans. Audio Speech Lang. Process. **23**, 461–471 (2015)
8. Gambhir, M., Gupta, V.: Recent automatic text summarization techniques: a survey. Artif. Intell. Rev. **47**(1), 1–66 (2016). https://doi.org/10.1007/s10462-016-9475-9
9. Glavas, G., Vulic, I.: Generalized tuning of distributional word vectors for monolingual and cross-lingual lexical entailment. In: 57th Conference of the Association for Computational Linguistics (ACL), pp. 4824–4830 (2019)
10. Gupta, A., Lebret, R., Harkous, H., Aberer, K.: Taxonomy induction using hypernym subsequences. In: Conference on Information and Knowledge Management (CIKM), pp. 1329–1338 (2017)
11. Harris, Z.S.: Distributional structure. Word **10**(2–3), 146–162 (1954)
12. Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In: 14th Conference on Computational Linguistics (COLING), pp. 539–545 (1992)
13. Jackendoff, R.: Foundations of Language: Brain, Meaning, Grammar, and Evolution. Oxford University Press, Oxford (2002)
14. Jiménez, S., González, F.A., Gelbukh, A.F., Dueñas, G.: Word2set: WordNet-based word representation rivaling neural word embedding for lexical similarity and sentiment analysis. IEEE Comput. Intell. Mag. **14**, 41–53 (2019)

15. Kathuria, N., Mittal, K.: A comprehensive survey on query expansion techniques, their issues and challenges. Int. J. Comput. Appl. **168**, 17–20 (2017)
16. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. In: 3rd International Conference on Learning Representations (ICLR) (2015)
17. Kozareva, Z., Hovy, E.: A semi-supervised method to learn and construct taxonomies using the web. In: Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1110–1118 (2010)
18. Ledig, C., Shi, W., Bai, W., Rueckert, D.: Patch-based evaluation of image segmentation. In: Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3065–3072 (2014)
19. 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 (HLT-NAACL), pp. 970–976 (2015)
20. Lézoray, O.: Patch-Based mathematical morphology for image processing, segmentation and classification. In: Battiato, S., Blanc-Talon, J., Gallo, G., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2015. LNCS, vol. 9386, pp. 46–57. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25903-1_5
21. 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)
22. Lu, P., Ji, L., Zhang, W., Duan, N., Zhou, M., Wang, J.: R-VQA: learning visual relation facts with semantic attention for visual question answering. In: 24th International Conference on Knowledge Discovery and Data Mining (KDD), pp. 1880–1889 (2018)
23. MIkołajczak-Matyja, N.: The associative structure of the mental lexicon: hierarchical semantic relations in the minds of blind and sighted language users. Psychol. Lang. Commun. **19**, 1–18 (2015)
24. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to WordNet: an on-line lexical database. Int. J. Lexicogr. **3**(4), 235–244 (1990)
25. 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)
26. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: bringing order to the web. Technical report 1999–66, Stanford InfoLab, November 1999
27. 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)
28. 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)
29. Roller, S., Kiela, D., Nickel, M.: Hearst patterns revisited: automatic hypernym detection from large text corpora. In: 56th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 358–363 (2018)
30. 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 (LREC), pp. 4557–4564 (2016)

31. 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)
32. 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 (ACL), pp. 2389–2398 (2016)
33. 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)
34. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems (2017)
35. Vulic, I., Mrksic, N.: Specialising word vectors for lexical entailment. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), pp. 1134–1145 (2018)
36. Vulic, I., Mrksic, N., Reichart, R., Séaghdha, D.Ó., Young, S.J., Korhonen, A.: Morph-fitting: fine-tuning word vector spaces with simple language-specific rules. In: 55th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 56–68 (2017)
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. Wang, Y.: On cognitive foundations of creativity and the cognitive process of creation. Int. J. Cogn. Inform. Nat. Intell. **3**, 1–18 (2009)
39. 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)