Recognizing Semantic Relations: Attention-Based Transformers vs. Recurrent Models

Dmitri Roussinov[1](✉), Serge Sharoff[2], and Nadezhda Puchnina[3]

[1] University of Strathclyde, 16 Richmond Street, Glasgow G1 1XQ, UK
dmitri.roussinov@strath.ac.uk

[2] University of Leeds, Leeds LS2 9JT, UK
s.sharoff@leeds.ac.uk

[3] University of Tallinn, Narva maantee 25, 10120 Tallinn, Estonia
np486061@tlu.ee

Abstract. Automatically recognizing an existing semantic relation (such as "is a", "part of", "property of", "opposite of" etc.) between two arbitrary words (phrases, concepts, etc.) is an important task affecting many information retrieval and artificial intelligence tasks including query expansion, common-sense reasoning, question answering, and database federation. Currently, two classes of approaches exist to classify a relation between words (concepts) X and Y: (1) path-based and (2) distributional. While the path-based approaches look at word-paths connecting X and Y in text, the distributional approaches look at statistical properties of X and Y separately, not necessary in the proximity of each other. Here, we suggest how both types can be improved and empirically compare them using several standard benchmarking datasets. For our *distributional* approach, we are suggesting using an attention-based transformer. While they are known to be capable of supporting knowledge transfer between different tasks, and recently set a number of benchmarking records in various applications, we are the first to successfully apply them to the task of recognizing semantic relations. To improve a *path-based* approach, we are suggesting our original neural word path model that combines useful properties of convolutional and recurrent networks, and thus addressing several shortcomings from the prior path-based models. Both our models significantly outperforms the state-of-the-art within its type accordingly. Our transformer-based approach outperforms current state-of-the-art by 1–12% points on 4 out of 6 standard benchmarking datasets. This results in 15–40% error reduction and is closing the gap between the automated and human performance by up to 50%. It also needs much less training data than prior approaches. For the ease of re-producing our results, we make our source code and trained models publicly available.

1 Introduction

During the last few years, Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) have resulted in major breakthroughs and are behind

J. M. Jose et al. (Eds.): ECIR 2020, LNCS 12035, pp. 561–574, 2020.
https://doi.org/10.1007/978-3-030-45439-5_37

the current state-of-the-art algorithms in language processing, computer vision, and speech recognition [9]. Meanwhile, modeling higher level abstract knowledge still remains a challenging problem even for them. This includes classification of semantic relations: given a pair of concepts (words or word sequences) to identify the best semantic label to describe their relationship. The possible labels are typically "is a", "part-of", "property-of", "made-of", etc. This information is useful in many applications. For example, knowing that *London* is a *city* is needed for a Question Answering system to answer the question *What cities does the River Thames go through?* Information retrieval benefits from query expansion with more specific words, e.g. *transportation disasters* → *railroad disasters*. For the task of database federation, an attribute in one database (e.g. with values *France, Germany*, and *UK*) often needs to be automatically matched with an attribute called *country* in another database. Knowing the semantic relations allows large-scale knowledge base construction [11,23,33], automated inferencing [6,29], query understanding [31], post-search navigation [7], and personalized recommendation [34]. The connection between word meanings and their usage is prominent in the theories of human cognition [12] and human language acquisition [2]. While manually curated dictionaries exist, they are known to be out-of-date, not covering specialized domains, designed to be used by people, and exist for only a few well resourced languages (English, German, etc.). Therefore, here we are interested in methods for automated discovery (knowledge acquisition, taxonomy mining, etc.). As our Sect. 2 elaborates, this problem has been a subject of extensive exploration for more than three decades. Our results here suggest that *knowledge transfer*, that was recently demonstrated to be useful for the other tasks, can be also successfully applied to recognizing semantic relations leading to substantial performance improvements and needing much less training data.

The automated approaches to detecting semantic relations between concepts (words or phrases) can be divided into two major groups: (1) path-based and (2) distributional methods. *Path-based approaches* (e.g. [25]) look for certain patterns in the joint occurrences of words (phrases, concepts, etc.) in the corpus. Thus, every word pair of interest *(x,y)* is represented by the set of *word paths* that connect x and y in a raw text corpus (e.g. Wikipedia). *Distributional approaches* (e.g. [30]) are based on modeling the occurrences of each word, x or y, separately, not necessary in the proximity to each other. Our goal here is to *improve and compare both classes of approaches.*

Attention-based transformers (e.g. [28]) have been recently shown more effective than convolutional and recurrent neural models for several natural text applications, leading to new state-of-the-art results on several benchmarks including GLUE, MultiNLI, and SQuAD [4,8]. At the same time, we are not aware of any applications of attention-based transformers to the task of recognizing semantic relations, so we are the first to successfully apply them to this task. Thus, our contributions are as follows:

(1) We develop a novel neural path-based model that combines useful properties of convolutional and recurrent networks. Our approach resolves several

shortcomings of the prior models within that type. As a result, it outperforms the state-of-the art path-based approaches on 3 out of 6 well known benchmarking datasets, and on par on the other 3. (2) Our distributional approach worked better than our neural path-based model and outperformed current state-of-the-art by 1–12% points (15–40% error reduction) on 4 out of 6 (same) standard datasets, and on par on the remaining 2. (3) We show that the datasets that are not improved are those where we have already reached the human performance. (4) We illustrate that even our best model still has certain limitations which are not always revealed by the standard datasets.

We make our code and data publicly available.[1] The next section overviews the prior related work. It is followed by the description of the models, followed by our empirical results.

2 Related Work

The approaches to automatically classifying semantic relations between words can be divided into two major groups: (1) path-based and (2) distributional. *Path-based approaches* look for certain patterns in the joint occurrences of words (phrases, concepts, etc.) in some validation text corpus. Thus, every word pair of interest *(x,y)* is represented by the set of *word paths* that connect x and y in a raw text corpus (e.g. Wikipedia). The earliest *path-based* approach is typically attributed to "Hearst Patterns" [5] – a set of 6 regular expressions to detect "is-a" relations (e.g. *Y such as X*). Later works successfully involved trainable templates and larger texts (e.g. [18,27]). However, a major limitation in relying on patterns in the word paths is the sparsity of the feature space [14]. Distributed representation do not have such limitations, thus with deep neural representations ("embeddings", e.g. [13,17]) becoming popular, a number of successful models were developed that used word embeddings as features (concatenation, dot product or difference) and surpassed the path-based methods in performance [15,20], to the point that *the path-based approaches were perceived not be adding anything to the distributional ones.*

However, [10] noted that supervised distributional methods tend to perform "lexical memorization:" instead of learning a relation between the two terms, they learn an independent property of a single term in the pair. For example, if the training set contains pairs such as *(dog, animal)*, *(cat, animal)*, and *(cow, animal)*, the algorithm learns to classify any new *(x, animal)* pair as true, regardless of x. Shwartz et al. [24,25] (one of our baselines) successfully combined distributional and path-based approaches to improve the state-of-the performance, and thus proving that path-based information is also crucial for that. In their approach, each word path connecting a pair of concepts (X, Y) is mapped by an RNN into a *context vector*. Those vectors are averaged across all existing paths and fed to a two-layer fully connected network.

There have been several related studies following [24]: [26] did extensive comparison of supervised vs. unsupervised approaches to detecting "is-a" relation.

[1] https://github.com/dminus1/meta-cats.

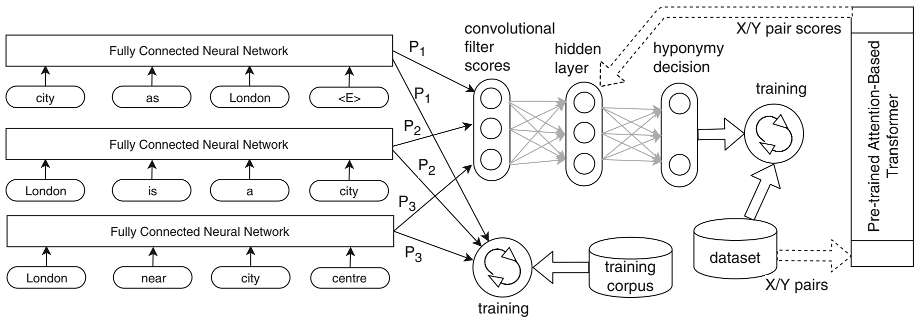
Fully Connected Neural Network
city
as
London
<E>
Fully Connected Neural Network
London
is
a
city
Fully Connected Neural Network
London
near
city
centre
P1
P2
P3
convolutional filter scores
hidden layer
hyponymy decision
X/Y pair scores
training
Pre-trained Attention-Based Transformer
dataset
X/Y pairs
training corpus
training

Fig. 1. Our path-based neural approach to semantic relationship classification.

[32] looked at how additional word paths can be predicted even if they are not in the corpus [19] also looked at "is-a" relation and confirmed the importance of modeling word paths in addition to purely distributional methods. Still, the results in [24,25] remain unsurpassed within the class of *word-path models.* Our baseline for distributional approaches is [30], who suggested using hyperspherical relation embeddings and improved the results of [24] on 3 out of 4 datasets.

3 Compared Models for Semantic Relations

3.1 Path-Based Neural Model

Intuitive Description. Our proposed path-based neural model combines useful properties of convolutional and recurrent networks, while resolving several shortcomings of the current state-of-the-art model [25] as we explain below. Figure 1 presents an informal intuitive illustration. We jointly train our semantic classification along with an unsupervised language modeling (LM) task.

The output of LM is the probability of occurrence of any input word sequence. *We use some of those probabilities as features for our relation classification model.* Inspired by the success of convolutional networks (CNNs), we use a fixed set of trainable filters (also called *kernels*), which learn to respond highly to certain patterns that are indicative of specific semantic relations. For example, a specific filter f_i can learn to respond highly to *is a* (and similar) patterns. At the same time, our recurrent LM may suggest that there is a high probability of occurrence of the sequence *green is a color* in raw text corpus. Combining those two facts suggests that *green* belongs to the category *color* (true *is-a* relation between them). Figure 1 shows only three convolutional filters (and the probabilities of the sequences P_1, P_2, P_3), while in our current study we used up to 16.

Thus, the LM probabilities act as approximate ("soft") pattern matching scores: (1) similar patterns receive similar scores with the same filter and (2) similar filters produce similar scores for the same pattern. LM also reduces the need for using many filters as explained by the following intuitive example: While training, LM can encounter many examples of sequences like *green is a popular* *color* and *green is a relaxing color.* By modeling the properties of a language, LM learns that removing an adjective in front of a noun does not normally result in a large drop of the probability of occurrence, so the sequence *green is a color* also scores highly even if it never occurs in the corpus.

Since the current state-of-the art path-based approach [25] aggregates the word paths connecting each target pair by averaging the context vectors representing all the paths, we believe their approach has two specific drawbacks that our approach does not: (1) when averaging is applied, the different occurrences of word patterns are forced to compete against each other, so the more rare occurrences can be dominated by more common ones and their impact on classification decision neglected as a result. By using LM we avoid facing the question how to aggregate the context vectors representing each path existing in the corpus. (2) The other relative strength of our approach over the baseline comes from the fact that our model does not "anonymize" the word paths unlike [25], which uniformly uses "x" and "y" for the path ends regardless of which words the target pair (x,y) actually represents. Without the use of LM, this anonymizing is unavoidable to generalize to the previously unseen (x,y) pairs, but it also misses the opportunity for the model to transfer knowledge from similar words.

Formal Definitions. Language Model (LM) is a probability distribution over sequences of words: $p(w_1, ..., w_m)$, where $w_1, ..., w_m$ is any arbitrary sequence of words in a language. We train LM jointly with our semantic relation classification task by minimizing cross-entropy costs, equally weighted for both tasks. As nowadays de-facto standard for a LM, we use a recurrent neural network (specifically a GRU variation [3], which works as well as LSTM while being faster to train). Thus, the probability of a word w_m in the language to follow a sequence of words $w_1, ..., w_{m-1}$ is determined by using the RNN to map the sequence $w_1, ..., w_{m-1}$ into its context vector:

$$\overrightarrow{v}_{w_1,...,w_{m-1}} = \mathrm{RNN}(w_1, ..., w_{m-1}) \tag{1}$$

and then applying a linear mapping and the softmax function:

$$\begin{aligned} p(w_m|w_1, ..., w_{m-1}) = \\ \mathrm{softmax}\left(W \cdot \overrightarrow{v}_{w_1,...,w_{m-1}} + b\right) \end{aligned} \tag{2}$$

where W is a trainable matrix, b is a trainable bias, and *softmax* is a standard function to scale any given vector of scores to probabilities.

As any typical neural LM, our LM also takes distributed representations of words as inputs: all the words are represented by their trainable *embedding* vectors $v_1, ..., v_m$.[2] This is important for our model and allows us to *treat LM as a function defined over arbitrary vectors* $p(v_m|v_1, ..., v_{m-1})$ *rather than over words.*

To classify semantic relations, we only look at the word paths that connect the target word pairs. Thus, we only make use of probabilities of the form

[2] We deliberately do not use the arrow over the word vectors to simplify the notation.

$p(v_y|v_x, v_1, ..., v_k)$, where (x, y) is one of the *target pairs* of words - those in the dataset that are used in training or testing the semantic relations, (v_x, v_y) are their embedding vectors. The sequence of vectors $v_1, ..., v_k$ defines a trainable *filter*, and k is its size. While vectors $v_1, ..., v_k$ have the same dimensions as the word embeddings, they are additional parameters in the model that we introduce. They are trained with the other ones (word embeddings + RNN matrices + the decision layer) by back propagation. It is possible since due to the smoothness of a neural LM, the entire model is differentiable.

Thus, we formally define the score of each of our convolutional filters (kernels) the following way:

$$f_i = \log p(v_y|v_x, v_1^i, ..., v_k^i) \tag{3}$$

where $p()$ is determined by our language model as the probability of the word with the embedding vector v_y to follow the sequence of words with the vectors $v_x, v_1^i, ..., v_k^i$. We apply *log* in order to deal with high variation in the orders of magnitude of $p()$. Finally, we define the vector of filter scores by concatenating the individual scores: $\overrightarrow{f} = [f_1, f_2, f_3, ..f_N]$, where N is the total number of filters (16 in our study here).

Filter scores $\overrightarrow{f}$ are mapped into a semantic relation classification decision by using a neural network with a single hidden layer. Thus, we define:

$$\overrightarrow{h_1} = \tanh(W_2 \cdot \overrightarrow{f} + b_2) \tag{4}$$

where W_2 is a trainable matrix and b_2 is a trainable "bias" vector. The classification decision is made based on the output activations:

$$c = \operatorname{argmax}(W_3 \cdot \overrightarrow{h_1} + b_3) \tag{5}$$

where W_3 and b_3 are also trainable parameters. As traditional with neural networks, we train to minimize the cross-entropy cost:

$$\mathit{cost} = -\log((\operatorname{softmax}(W_3 \cdot \overrightarrow{h_1} + b_3))[c_l]) \tag{6}$$

where c_l is the correct (expected) class label. We used stochastic gradient descent for cost minimization.

3.2 Distributional Model: Attention-Based Transformer

The diagram on Fig. 2 illustrates how attention-based transformer [28] operates. Instead of recurrent units with "memory gates" essential for RNN-s, attention-based transformers use additional word positional embeddings which allows them to be more flexible and parallelizable than recurrent mechanisms which have to process a sequence in a certain direction. The conversions from the inputs to the outputs are performed by several layers, which are identical in their architecture, varying only in their trained parameters. In order to obtain the vectors on the layer above, the vectors from layer immediately below are simply weighted and

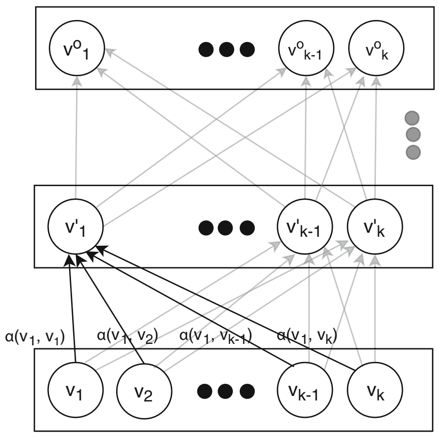

Fig. 2. Attention-based transformer used in our distributional approach to semantic relationship classification.

added together. After that, they are transformed by a standard nonlinearity function. We use *tanh*:

$$\overrightarrow{v_i}' = \tanh(W \cdot \sum_{t=1}^{k} \alpha_t \overrightarrow{v_t}) \quad (7)$$

here, $\overrightarrow{v_i}'$ is the vector in the i-th position on the upper layer, $\overrightarrow{v_t}$ is the vector in the t-th position on the lower layer, W is a trainable matrix (same regardless of i but different at different layers), and α_t is a trainable function of vectors $\overrightarrow{v_i}$ and $\overrightarrow{v_t}$, such as the weights for all $\overrightarrow{v_t}$ add up to 1. We use a scaled dot product of the vectors $\overrightarrow{v_i}$ and $\overrightarrow{v_t}$:

$$\alpha_t = \overrightarrow{v_i} \cdot W' \cdot \overrightarrow{v_t} \quad (8)$$

where W' is a trainable matrix (also same regardless of i and t at the same layer but different at different layers). The normalization to 1 is accomplished by using a *softmax* function.

This mechanism allows rich vector representations to be formed at the highest layers that can capture the entire content of a word sequence (e.g. a sentence or a word pair) so it can be effectively used for any AI applications such as text classification or generation. As it is commonly done with the transformers, we make our output classification decision based on the first vector on the top level. We do not use a hidden layer here, so we apply our formula 5 above to h_1 defined as the following:

$$\overrightarrow{h_1} = \overrightarrow{v_0^u} \quad (9)$$

where $\{\overrightarrow{v_t^u}\}$ is the vector sequence produced by the transformer for the top level.

Table 1. The relation types and statistics in each dataset.

Dataset	Dataset relations	#inst.	#uniq. X	#uniq. Y
Hypenet Lexical	is a	20335	16044	5148
Hypenet Random	is a	49475	38020	12600
K& H+N	is a, part of	57509	1551	16379
BLESS	is a, part of, event, attribute	26546	201	8089
ROOT09	is a	12762	1218	3436
EVALution	is a, part of, attribute, opposite, made of	7378	1631	1497

4 Empirical Evaluation

4.1 The Datasets

Table 1 summarizes general statistics of the datasets. We used the same datasets as our baselines: the first two are from [25] and were built using a similar methodology: the relations used in them have been primarily taken from various sources including WordNet, DBPedia, Wikidata and Yago. Thus, their x-s are primarily named entities (places, films, music albums and groups, people, companies, etc.). The important difference is that in order to create the split between training, testing and validation sets for HypeNet Lexical, the lexical separation procedure was followed [10], so that there is no overlap in words (neither x nor y) between them. This reduces "lexical memorization" effect mentioned above. The last four datasets are from [24], which originate from various preceding studies: K&H+N [15], BLESS [1], ROOT09 [20], EVALution [21]. Most of the relations for them were also taken WordNet. BLESS dataset also contains *event* and *attribute* relations, connecting a concept with a typical activity/property, e.g. *(alligator, swim)* and *(alligator, aquatic)*. EVALution dataset contains the largest number of semantic relations including antonyms, e.g. *(good, bad)*. To make our comparison more direct, we used exactly the same splits into training, development (validation) and testing subsets as in the baselines. We also used exactly the same word paths data, as it is made publicly available by the authors.

4.2 Experimental Setups

Since we sought to keep the number of hyper-parameters to the minimum, we set the word embedding size, the RNN context vector size, and the hidden layer size to be the same within all our path-based models. We tested their values in the {50,100,500,1000} set. This size is the only hyper-parameter that was varied in our experiments. We used the static learning rate of 0.01. As it is commonly done, we report the results computed on the test sets with the hyper-parameter and the number of training iterations that maximize the F_1 scores on the validation sets,

Table 2. F_1 scores of our tested models compared to the state-of-the-art baselines. Datasets: HyperNet Lexical (HL), Hypenet Random (HR), BLESS (B), ROOT09 (R), EVALution(E).

Model	HL	HR	K&H	B	R	E
Prior Shwartz et al. (2016)	0.660	0.890	0.983	0.889	0.788	0.595
Word path models:						
Shwartz et al. (2016)	0.700	0.901	0.985	0.893	0.814	0.600
Our neural model	0.740	0.899	**0.990**	0.927	0.832	0.602
Distributional:						
Wang et al. (2019)	N/A	N/A	**0.990**	0.938	0.861	0.620
Our transformer-based	**0.821**	**0.905**	0.987	**0.950**	**0.905**	**0.701**
Human	0.90	0.90	0.98	0.96	0.95	0.82

thus using exactly the same metrics and procedures as were used to obtained the baseline results: scikit-learn [16] with the "weighted" set-up, which computes the metrics for each relation, and reports their average, weighted by support (the number of true instances for each relation). For HypeNet datasets, that was accordingly set to "binary". We also verified through personal communications with the authors of [24] that our metrics are numerically identical for the same sets of predicted labels.

For our path-based models, all the trainable parameters were initialized by a normal distribution around 0 average and standard deviation of 1. We used the same transformer architecture and hyper-parameters as in [4] (BERT monolingual English uncased version) which has 12 layers and the output vector size of 768, resulting in the total number of trainable parameters of 110 million. As it is commonly done when using a pre-trained transformer, we initialize our weights to those that were already trained by [4] for a language model and next sentence prediction tasks on a copy of English Wikipedia text and the BookCorpus. For consistency with the data used during pre-training, we add the same special markers before, between and after our input word sequences x and y.

4.3 Comparing Against the Baselines

Table 2 presents our results. For additional comparison, we also include "Before Baseline" row, which lists the baselines used in [24,25]. For HypeNet Random and Evalution datasets, we put the larger values that we obtained in our re-implementation of the distributional methods that they used rather than their reported values. The following can be observed:

(1) Our **neural word path model** has been able to improve the state-of-the-art on three (3) out of six (6) datasets: Hypenet L, Bless and Root09. The differences are statistically significant at the level of .01. On the remaining three (3) datasets (HypeNet Random, K&H+N and Evalution), our results

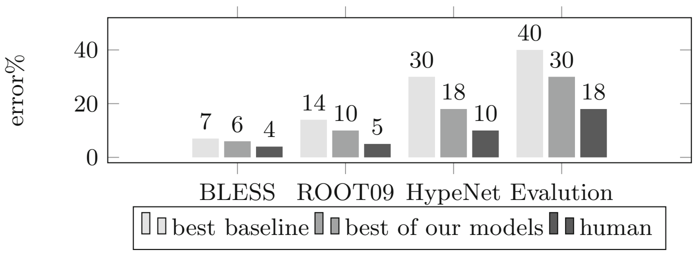

Fig. 3. Visual illustration of the error reduction relatively to the baseline and human performance on the tasks.

are the same as with the baseline performance (no statistically significant difference at the level .05). The baseline did not improve on those datasets over the prior work either. The scores for HypeNet Random and K&H+N are already high due to "lexical memorization" mentioned above. Since the compared models used exactly the same data, the obtained results clearly suggest that *our neural model is better* than the current state-of-the-art word-path model [24,25].

(2) Our **transformer-based** model has also demonstrated tangible gains over state-of-the-art baselines regardless of the class of the approach (both path-based and distributional) on four (4) out of six (6) datasets by 1–12% points (15–40% error reduction). Those differences are statistically significant at the level of .01. There are no statistically significant differences on HypeNet Random and K&H+N. This suggests that *an attention-based transformer is a very powerful mechanism for modeling semantic relations.* Although they have been shown to be very effective in many other applications where knowledge transfer between tasks is essential, this is the first study that has used them for semantic relations.

(3) On four (4) out of six (6) datasets, our **distributional model worked better** than our neural word path model. The differences are statistically significant at the level of .01. There are no statistically significant differences on the remaining two.

We estimated the human performance on our datasets by giving 100 randomly selected word pairs to 3 independent graders, who were allowed to look up the meanings online (last row). It can be seen that the state-of-the-art approaches have already achieved the human level on the datasets where no improvement was detected (HypeNet Random and K&H+N), so this may explain why our approaches did not substantially improve them any further. Fig. 3 illustrates the effect of error reduction on the four datasets on which our approaches improved the state-of-the-art. For comparison, we plot the semantic relation classification error calculated as $100 - F_1$ score rounded to the nearest integer. It can be seen that our approaches have approximately reduced the errors on those "unsolved" datasets half-way from the baseline to the human level. We believe that this result is truly remarkable!

Table 3. Average Precision (AP) scores of our tested models compared to other recent strong baselines on binary category verification ("is-a" relation only).

Model	BLESS	EVALution
Chang et al. 2018	0.186	0.364
Roller et al. 2018	0.760	0.480
Nguyen et al. 2017	0.454	0.538
Yin and Roth 2018	0.595	0.623
Our path-based	0.939	0.603
Our transformer	**0.986**	**0.739**

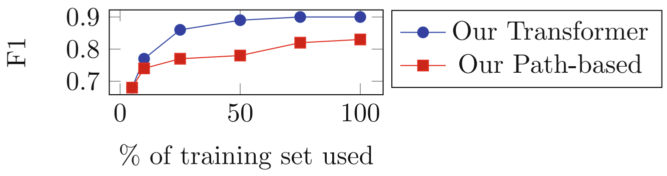

Fig. 4. Using only portion of Root dataset for training.

For additional comparison, we also include our results along with the results of other recent works that looked at semantic relations classification even though those works did not claim to exceed the state-of-the-art approaches presented in [24,25]. We report the metric of Average Precision (AP) used in those studies and the results on the two datasets (Bless and Evalution) also commonly used in them. We did not use the other datasets for comparison since they are much smaller and relying on manual part of speech tags (nouns, verbs, adjectives etc.). As Table 3 illustrates, our path-based model outperforms all but one, and our transformer-based model sizably exceeds all of those results reported.

We have also tested the influence of training size on the model by comparing its performance with 5%, 10%, 25%, 50% and 75% of randomly selected training subsets. Due to the size limit, we show only the results on Root09 (Fig. 4). The results suggest the importance of the dataset size and the possibility of further improvements when more training data is available for the path-based. At the same time, out transformer-based model needs much less training to reach its top possible performance. We also verified that all the components of our path-based model here are essential to exceed the baselines, specifically: using a hidden layer, using all the available word paths, using all 16 filters. Larger number of filters did not result in any gains, but increased the training time.

We also tried to play an adversarial role and fed more challenging pairs to the trained models to see when they are starting to fail. Our attention-based transformer model trained for HypeNet Lexical dataset (named entities mostly) erroneously classified all the 100 examples created by combining random general words and the word "air" (e.g. "car air", "circle air", "new air") as "airline."

It also erroneously classified all the 30 correct airline names that we tried as "airports" in addition to correctly classifying them as "airline." The proportion of correct airline names classified as "recording label" was 60%, which is lower than for the correct category, but still alarmingly high. Meanwhile, general words (like "car", "book", "new", etc.) are very rarely classified as members of any categories in this dataset since the model correctly sees that they are not named entities. Those observations suggest that what the transformer actually learns for this datasets is to use the combined properties of a word sequence (n-gram) to check if it can possibly be a named entity, and then if it *topically* fits the category (e.g. "aviation" in general). Those two conditions are sufficient to make a positive classification and to obtain high scores since very few test categories in the dataset are closely related (e.g. "airport" and "airline"). While our neural path models don't make such mistakes, their mistakes are primarily due to no word paths existing between the candidates in the training corpus, which was already noted in the related prior work. This suggests that a combination of those two approaches may provide additional gains over each. We have left more formal exploration of those observations for future studies.

5 Conclusions

We have considered the task of automatically recognizing semantic relations between words (phrases, concepts, etc.) such as "is a", "part of", "property of", "opposite of" etc., which is an important task affecting many applications including knowledge base construction, inference, query understanding, personalized recommendation and post-search navigation. Using six standard datasets, we have demonstrated that both distributional and word path state-of-the-art approaches can be tangibly improved. Out of those two approaches that we suggested, the transformer-based distributional approach worked significantly better. It has decreased the gap between the current strong baselines and human performance by roughly 50% for those datasets that still had room for improvement. We are not aware of any other work applying a pre-trained attention-based transformer (ABT) for this task. Since ABT-s are currently known to be the first practically useful mechanism for knowledge transfer between natural language tasks, our work paves the way to making knowledge transfer to be a default feature in any modern NLP tool.

It will also lead to integrating training of the transformer with the semantic classification task on a deeper level, which can be accomplished by customizing its pre-training (weight-initialization) algorithm to include word semantic information available from existing taxonomies, which we are planning to undertake in future, along with experimenting with cross-lingual knowledge transfer (e.g. [22]), when a model uses English data to predict semantic relations in other, less resourced, languages.

References

1. Baroni, M., Lenci, A.: How we BLESSed distributional semantic evaluation. In: Proceedings of 2011 Workshop on Geometrical Models of Natural Language Semantics (2011)
2. Bybee, J.L., Beckner, C.: Usage-based theory. In: The Oxford Handbook of Linguistic Analysis (2015)
3. Cho, K., van Merrienboer, B., Gulcehre, C., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: 55th Annual Meeting of the Association for Computational Linguistics (2014)
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
5. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: ACL 1992 (1992)
6. Inkpen, D., Zhu, X., Ling, Z.H., Chen, Q., Wei, S.: Neural natural language inference models enhanced with external knowledge. In: ACL (2018)
7. Keller, M., Mühlschlegel, P., Hartenstein, H.: Search result presentation: supporting post-search navigation by integration of taxonomy data. In: WWW Conference (2013)
8. Lample, G., Conneau, A.: Cross-lingual language model pretraining. arXiv preprint arXiv:1901.07291 (2019)
9. LeCun, Y., Bengio, Y., Hinton, G.E.: Deep learning. Nature **521**, 7553 (2015)
10. Levy, O., Remus, S., Biemann, C., Dagan, I.: Do supervised distributional methods really learn lexical inference relations? In: Proceedings of NAACL-HLT (2015)
11. Mahdisoltani, F., Biega, J., Suchanek, F.M.: Yago3: a knowledge base from multilingual wikipedias. In: CIDR (2015)
12. Mikoajczak-Matyja, N.: The associative structure of the mental lexicon: hierarchical semantic relations in the minds of blind and sighted language users. In: Psychology of Language and Communication (19) (2015)
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS 2013 (2013)
14. Nakashole, N., Weikum, G., Suchanek, F.: PATTY: a taxonomy of relational patterns with semantic types. In: 2012 Joint Conference EMNLP and CoNLL (2012)
15. Necsulescu, S., Mendes, S., Jurgens, D., Bel, N., Navigli, R.: Reading between the lines: overcoming data sparsity for accurate classification of lexical relationships. In: SEM 2015 (2015)
16. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
17. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: EMNLP 2014 (2014)
18. Riedel, S., Yao, L., McCallum, A., Marlin, M.B.: Relation extraction with matrix factorization and universal schemas. In: NAACL-HLT 2013 (2013)
19. Roller, S., Kiela, D., Nickel, M.: Hearst patterns revisited: automatic hypernym detection from large text corpora. In: ACL 2018 (Short Paper) (2018)
20. Santus, E., Lenci, A., Chiu, T.S., Lu, Q., Huang, C.R.: Nine features in a random forest to learn taxonomical semantic relations. In: LREC 2016 (2016)

21. 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: Proceedings of the 4th Workshop on Linked Data in Linguistics: Resources and Applications (2015)
22. Sharoff, S.: Finding next of kin: cross-lingual embedding spaces for related languages. Nat. Lang. Eng. **26**(2), 163–182 (2019)
23. Shen, J., et al.: Yago3: a knowledge base from multilingual wikipedias. In: KDD (2018)
24. Shwartz, V., Dagan, I.: Path-based vs. distributional information in recognizing lexical semantic relations. In: COLING 2016 (2016)
25. Shwartz, V., Goldberg, Y., Dagan, I.: Improving hypernymy detection with an integrated path-based and distributional method. In: ACL 2016 (2016)
26. Shwartz, V., Santus, E., Schlechtweg, D.: Hypernyms under siege: linguistically-motivated artillery for hypernymy detection. In: EACL 2017 (2017)
27. Snow, R., Jurafsky, D., Ng, A.Y.: Learning syntactic patterns for automatic hypernym discovery. In: NIPS 2004 (2004)
28. Vaswani, A., et al.: Attention is all you need. In: NIPS 2017 (2017)
29. Vulic, I., Gerz, D., Kiela, D., Hill, F., Korhonen, A.: Hyperlex: a large-scale evaluation of graded lexical entailment. Comput. Linguist. **43**(4), 781–835 (2017)
30. Wang, C., He, X., Zhou, A.: Spherere: distinguishing lexical relations with hyperspherical relation embeddings. In: ACL 2019 (2019)
31. Wang, Z., Zhao, K., Wang, H., Meng, X., Wen, J.R.: Query understanding through knowledge-based conceptualization. In: IJCAI (2015)
32. Washio, K., Kato, T.: Filling missing paths: modeling co-occurrences of word pairs and dependency paths for recognizing lexical semantic relations. In: NAACL-HLT 2018 (2018)
33. Wu, W., Li, H., Wang, H., Zhu, K.Q.: Probase: a probabilistic taxonomy for text understanding. In: SIGMOD (2012)
34. Zhang, Y., Ahmed, A., Josifovski, V., Smola, A.J.: Taxonomy discovery for personalized recommendation. In: WSDM (2014)