Unraveling Antonym's Word Vectors through a Siamese-like Network

Mathias Etcheverry
Republic University
Montevideo, Uruguay
mathiase@fing.edu.uy

Dina Wonsever
Republic University
Montevideo, Uruguay
wonsever@fing.edu.uy

Abstract

Discriminating antonyms and synonyms is an important NLP task that has the difficulty that both, antonyms and synonyms, contains similar distributional information. Consequently, pairs of antonyms and synonyms may have similar word vectors. We present an approach to unravel antonymy and synonymy from word vectors based on a siamese network inspired approach. The model consists of a two-phase training of the same base network: a pre-training phase according to a siamese model supervised by synonyms and a training phase on antonyms through a siamese-like model that supports the antitransitivity present in antonymy. The approach makes use of the claim that the antonyms in common of a word tend to be synonyms. We show that our approach outperforms distributional and pattern-based approaches, relaying on a simple feed forward network as base network of the training phases.

1 Introduction

Antonymy and synonymy are lexical relations that are crucial in language semantics. Antonymy is the relation between opposite words, (e.g. big-small) and synonymy refers to words with similar meaning (e.g. bug-insect). Detecting them automatically is a challenging NLP task that can benefit many others like textual entailment (Haghighi et al., 2005; Snow et al., 2006), machine translation (Bar and Dershowitz, 2010) and abstractive summarization (Khatri et al., 2018).

Hand crafted lexical databases, such as WordNet (Miller, 1995), have been built and maintained to be used in NLP and other fields containing antonyms, synonyms and other lexical semantic relations. However, its construction and maintenance takes a considerable human effort and it is difficult to achieve a broad coverage. Detecting antonyms automatically, relying on existent resources such as text, dictionaries and lexical databases is an active NLP research area.

In the last decade, the use and research concerning word vectors have increased rapidly. Word vectors rely on words co-occurrence information in a large corpus. The key idea behind word vectors is the distributional hypothesis that can be expressed as "the words that are similar in meaning tend to occur in similar contexts" (Sahlgren, 2008; Rubenstein and Goodenough, 1965). A variety of methods have been developed to train word vectors, such as skip-gram (Mikolov et al., 2013), GloVe (Pennington et al., 2014), FastText (Joulin et al., 2016) and ElMo (Peters et al., 2018). Word vectors are used widely in NLP, for example, a well-known use is in supervised learning, taking advantage of the expansion through words relatedness of the training data.

A main problem to discriminate antonymy automatically in a distributional unsupervised setting is that the oppositeness is not easily distinguishable in terms of the context distributions. In fact, pairs of antonyms are very similar in meaning. Antonyms are usable in the same contexts but leading to opposite meanings. Antonymy is said to have the paradox of simultaneous similarity and difference (Cruse, 1986), because antonyms are similar in almost every dimension of meaning except the one where they are opposite.

The paradox of simultaneous similarity and difference is notorious in word space models. The contexts of a word and its antonyms contexts usually are similar and therefore they have close vector representations[1].

[1] In fact, word space models may give similar representations to a broader range of related words, such as synonyms and hyponyms. Note the difference between the terms word similarity and word relatedness. While word similarity refers to similar words (synonyms), the concept of word relatedness

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3297–3307
Florence, Italy, July 28 - August 2, 2019.

Due to this paradox, word space models seem not suitable for antonymy detection. Then, a commonly used resource is the path of words connecting the joint occurrence of two candidate words (Nguyen et al., 2017). Path based approaches take profit of the fact that antonyms co-occur in the same context more than expected by chance (Scheible et al., 2013; Miller and Charles, 1991), so it is possible to obtain a significant amount of patterns.

In this paper, we claim that vector space models, despite giving close representations for synonyms and antonyms, contain subtle differences that allow to discriminate antonymy. In order to stick out those differences we propose a method based on a neural network model that takes account of algebraic properties of synonymy and antonymy. The model formulation is based on the transitivity of synonymy and the antitransitivity of antonymy, on the symmetry of both relations and on the reflexivity and irreflexivity of synonymy and antonymy, respectively. Moreover, the model exploits the property that two antonyms of the same word tend to be synonyms (Edmundson, 1967) (Figure 1). We use these properties to define a model based on siamese networks and a training strategy through antonyms and synonyms.

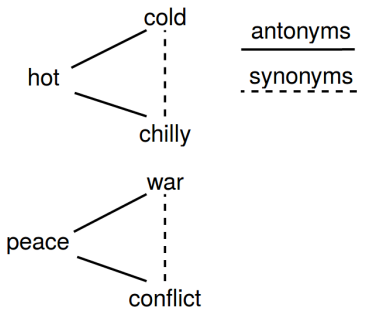

Figure 1: The antonyms of a same word tend to be synonyms.

We show that the presented approach gives surprisingly good results, even in comparison to models that use external information, such as dependency parsing, part-of-speech tagging or path patterns from a corpus. The introduced model is a way to learn any kind of antitransitive relations between distributed vectors. Antitransitivity may be suitable, for instance, to represent the relation of being adversary (Bonato et al., 2017). A different application of the presented approach could be in social networks in order to find out possible unknown enemies relying on a given set of known enmity and friendship links.

The rest of the paper is structured as follows: In Section 2 we present the previous work on antonymy detection. In Section 3 we describe the proposed approach. We start with some algebraic principles of synonymy and antonymy on which our approach relies. Then we describe siamese networks and how the learned transformation tends to induce an equivalence relationship, suitable for synonyms. In Section 3.3, we comment the unsuitability of siamese networks to deal with an antitransitive relationship like antonymy and we propose a variation of the original siamese network to do so. We refer to this network as a parasiamese network. Then, we argue that the same base network of a parasiamese model for antonymy can be pre-trained minimizing a siamese scheme on synonyms. Section 4 details the dataset, word vectors and the random search strategy carried out to find out an adequate hyperparameter configuration. In Section 5 we present the results and the behavior of the model. Finally, Section 6 contains the conclusion of this paper.

2 Related Work

Antonymy detection, and antonymy and synonymy discrimination, have been treated principally by two approaches: distributional and pattern-based. Distributional approaches refer to the use of word vectors or word's distributional information. Pattern-based are those that rely on patterns of joint occurrences of pair of words (such as "from X to Y") to detect antonymy. Due to the direction of this work, we will not extend on path-based approaches and we will give the most attention in this section to distributional approaches.

As we commented before, at first glance word vectors seem not suitable to discriminate antonymy from synonymy because pairs of antonyms correspond to similar vectors. Many research studies and experiments have focused on the construction of vector representations that deem antonymy.

Scheible et al. (2013) showed that the context distributions of adjectives allow to discriminate antonyms and synonyms if only words from certain classes are considered as context in the vector space mode. Hill et al. (2014) found that word vectors from machine translation models outperform

includes other semantic fields, like antonyms, hypernyms, co-hyponyms and specific relations (e.g. dog-bone).

those learned from monolingual models in word similarity. They suggest that vectors from machine translation models should be used on tasks that require word similarity information, while vectors from monolingual models are more suitable for word relatedness. Santus et al. (2014) proposed APAnt, an unsupervised method based on average precision of contexts intersections of two words, to discriminate antonymy from synonymy.

Symmetric patterns in corpus (e.g. X and Y) were used by Schwartz et al. (2015) to build word vectors and they showed that the patterns can be chosen so that the resulting vectors consider antonyms as dissimilar. Ono et al. (2015) proposed an approach to train word vectors to detect antonymy using antonymy and synonymy information from a thesauri as supervised data. A main difference between their approach and ours is that they did not rely on pre-trained vectors. They used distributional information jointly with the supervised information to train vectors through a model based on skip-gram. Also, Nguyen et al. (2016) integrated synonymy and antonymy information into the skip-gram model to predict word similarity and distinguish synonyms and antonyms.

More recently, Nguyen et al. (2017) distinguish antonyms and synonyms using lexico-syntactic patterns jointly with the supervised word vectors from Nguyen et al. (2016). To finish, (Vulić, 2018) obtain great performance injecting lexical contrast into word embeddings by terms of their ATTRACT-REPEL strategy.

3 Method

In this section we describe the proposed approach to discriminate antonymy and synonymy. It consists on a siamese networks inspired approach to magnify the subtle differences on antonyms that distinguish them from synonyms.

3.1 Algebra of synonymy and antonymy

In order to define and substantiate our approach we introduce an axiomatic characterization of antonymy and synonymy based on the work done by Edmundson (1967). Precisely, synonymy and antonymy are modeled as relations and a set of axioms is proposed. These axioms, as we are going to show, are essential to formulate our approach.

At first glance, synonymy and antonymy can be seen as binary relations between words. However, based on empirical results[2] Edmundson defined synonymy and antonymy as ternary relations in order to consider the multiple senses of the words, as follows:

$$xS_iy \equiv x \text{ synonym of } y \text{ according to sense } i$$
$$xA_iy \equiv x \text{ antonym of } y \text{ according to sense } i$$

Note that the senses of the words are represented in the relationship rather than in the words themselves. Each i (and therefor S_i and A_i) reflects a particular configuration of the senses of the words in the vocabulary, considering a unique sense for each word.

Firstly, synonymy is considered a reflexive, symmetric and transitive relationship. This is expressed by the following axioms:

$$\forall i \forall x (xS_ix) \tag{1}$$
$$\forall i \forall x \forall y (xS_iy \implies yS_ix) \tag{2}$$
$$\forall i \forall x \forall y \forall z (xS_iy \wedge yS_iz \implies xS_iz) \tag{3}$$

S_i is an equivalence relation for each fixed i and therefor it splits the set of words into equivalence classes. In the next section we show that this is suitable for siamese networks.

Antonymy is also a symmetric relation but it is irreflexive and antitransitive:

$$\forall i \forall x \neg (xA_ix) \tag{4}$$
$$\forall i \forall x \forall y (xA_iy \implies yA_ix) \tag{5}$$
$$\forall i \forall x \forall y \forall z (xA_iy \wedge yA_iz \implies \neg xA_iz) \tag{6}$$

So far, synonymy and antonymy are described separately. The following two axioms involve both relationships:

$$\forall i \forall x \forall y \forall z (xA_iy \wedge yA_iz \implies xS_iz) \tag{7}$$
$$\forall i \forall x \forall y \forall z (xA_iy \wedge yS_iz \implies xA_iz) \tag{8}$$

Axiom 7 is a refined version of the antitransitive property (axiom 6)[3]. Assuming that two words cannot be synonyms and antonyms simultaneously, it is direct to prove that axiom 7 implies axiom 6. We include axiom 6 for clarification purpose.

[2]Precisely, analyzing graphs of several sets of synonyms treated as a binary relation and the adjacency matrices associated with these graphs.

[3]In fact, the term antitransitivity is used in Edmundson article to refer axiom 7 instead of axiom 6

The right-identity, axiom 8, says that synonyms of an antonym of a word are also antonyms. Consequently, antonymy relation can be extended to operate between synonymy equivalence classes.

To introduce our model and the considered task setting, we simplify this definition enforcing a binary relation. We consider:

$$xRy \iff \exists i(xR_iy),$$

where R and R_i are S or A and S_i or A_i, respectively. This simplification encapsulates the multiple senses of the words and therefore it is suitable for word embeddings. However, the presented axioms may not be completely fulfilled under this simplification.

3.2 Synonymy and Siamese Networks

A siamese network is a model that receives two inputs an returns an output. A base neural network is applied to each input and the both outputs are measured using a vector distance function (see Figure 2). Usually, siamese networks are trained using a contrastive loss function. The complete model can be interpreted as a trainable distance function on complex data, like images, sound, or text. Siamese networks have been used in a variety of tasks such as sentence similarity (Chi and Zhang, 2018), palmprint recognition (Zhong et al., 2018) and object tracking (Bertinetto et al., 2016), among many others.

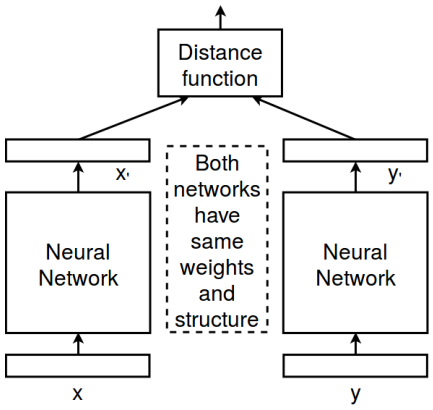

Figure 2: A siamese network model.

Consider a vocabulary V of words where we want to discriminate synonyms and a given word vector set for that vocabulary of dimension n. Then consider a neural network $F_\theta : \mathbb{R}^n \to \mathbb{R}^n$ with weights θ and the following contrastive loss function

$$L = \sum_{(x,y)\in P} d(F_\theta(x), F_\theta(y)) + \sum_{(x',y')\in N} max\{0, \alpha - d(F_\theta(x'), F_\theta(y'))\},$$

where $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^+$ is a vector distance function (e.g. $d = ||x - y||^2$), α is the threshold for the negative examples, and P and N are positive and negative example pairs, respectively. So P is a set of pairs of synonyms and N a set of pair of words that are not synonyms. We consider that each pair is already composed by the word vector of each word, this is convenient to simplify the notation. This model can be trained using a back-propagation based technique and the output vectors closer than a given threshold are classified as related.

It can be proved that the relation induced by a siamese network is reflexive and symmetric. Transitivity is a little more tricky. It is assured to be satisfied when the sum of the distances of the antecedent related pairs is below the threshold and, in every case, the distance of the transitive pairs is below the double of the threshold. Therefore, a siamese network is a reasonable approach for supervised synonymy detection.

3.3 Antonymy and Antitransitivity

While a siamese network seems a reasonable choice for supervised synonym detection, antonymy presents a really different scenario. Consider $F_{\theta*}$ as the base neural network in a siamese scheme and suppose that it is trained and working perfectly to discriminate pairs of antonyms. Consider also three words w_1, w_2, w_3 such that w_1 is antonym of w_2 and w_2 is antonym of w_3, then

$$F_{\theta*}(w_1) = F_{\theta*}(w_2),$$
$$F_{\theta*}(w_2) = F_{\theta*}(w_3)$$

hence, $F_{\theta*}(w_1) = F_{\theta*}(w_3)$ an therefore, w_1 and w_3 would be recognized as antonyms, violating axiom 6.

A siamese network induces a transitive relationship but antonymy is actually antitransitive. To model an antitransitive relation, we propose the following variation of the siamese network.

Let's consider F_θ and the model diagrammed in figure 3. It consists of a model that consumes two vectors with the same dimension and applies a base neural network once to one input and twice

to the other. The idea behind this scheme is that if two word are antonyms then the base network applied once in one word vector and twice in the other word vector, will return close vectors. It can be interpreted as one application of the base network takes to a representation of the equivalence class of the synonymy relation and the second application to a representation its opposite class in terms of antonymy.

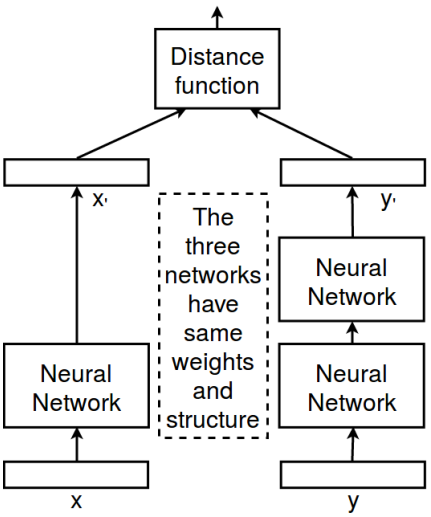

Figure 3: The proposed parasiamese network to discriminate antonymy.

Assume that $F_{\theta*}$ is trained and behaves perfectly on data according to the following loss function:

$$L_{ant} = \sum_{(x,y)\in P} d(F_\theta(x), F_\theta(F_\theta(y))) + \sum_{(x',y')\in N} max\{0, \alpha - d(F_\theta(x'), F_\theta(F_\theta(y')))\},$$

where P and N are positive and negative example pairs, respectively; α is the threshold for the negative examples, and d a distance function as in siamese network. Then, it can be seen that the relation induced fulfills the antitransitivity property if $F_{\theta*}(w) \neq F_{\theta*}(F_{\theta*}(w))$, which is expected since antonymy is an antireflexive relation.

Symmetry is not forced by definition but can be included in the loss function or by data, adding the reversed version of each pair in the dataset. The latter is the alternative chosen in this work.

3.4 Relaxed Loss Function

In order to classify a pair of words we rely in a threshold ρ. If the candidate pair obtains a distance (between its transformed vectors) below ρ, then it is classified as positive, otherwise as negative. So, it is not necessary to minimize the distance to 0 to classify it correctly. We propose to change the positive part of the contrastive loss function by

$$\sum_{(x,y)\in P} max(d(F_\theta(x), F_\theta(F_\theta(y))) - \rho\nu, 0)$$

where ν is a factor in $[0, 1]$ that states the importance given to ρ, the rest of the terms remains the same as in the previous section. If $\nu = 0$ then the original loss function is recovered. We consider $\nu = 1/2$ and we experimentally observe an improvement in results when this relaxed loss function is used.

3.5 Pre-training using synonyms

Consider $F_{\theta*}$ trained and perfectly working to detect pairs of antonyms using the parasiamese scheme presented in the previous section. Now, lets consider the word vectors w_1, w_2 and w_3 such that w_1 is antonym of w_2 and w_2 is antonym of w_3. According to the parasiamese loss function we have that,

$$F_{\theta*}(w_1) = F_{\theta*}(F_{\theta*}(w_2)),$$
$$F_{\theta*}(F_{\theta*}(w_2)) = F_{\theta*}(w_3).$$

This implies that $F_{\theta*}(w_1) = F_{\theta*}(w_3)$, suggesting to F the role of a siamese network. On the other hand, using axiom 7 we have that w_1 and w_3 tend to be synonyms, which, as we previously show, fits fine for siamese networks.

Using this result, we propose to pre-train F_θ, minimizing a siamese network on synonymy data as in Section 3.2, and then perform the parasiamese training to detect antonyms as described in Section 3.3. We use the same antonymy/synonymy dataset to pre-train and train the parasiamese network and we experimentally observe that this pre-training phase improves the performance of the parasiamese model.

4 Experiments

In this section we describe the setup details of the experiments performed using the presented approach. Here we give the complete information to reproduce the experiments performed. We describe the dataset, word vectors set used and the random search strategy used for the hyperparameter configuration.

Model	Adjective			Verb			Noun		
	P	R	F_1	P	R	F_1	P	R	F_1
Baseline (concat)	0.691	0.664	0.677	0.756	0.641	0.694	0.764	0.716	0.739
AntSynNet	0.763	0.807	0.784	0.743	0.815	0.777	0.816	**0.898**	**0.855**
Parasiam (regular loss)	0.735	0.804	0.768	0.815	0.894	0.853	0.786	0.857	0.820
Parasiam (no pre-train)	0.764	0.848	0.804	0.825	0.892	0.857	0.787	0.849	0.817
Parasiam (ElMo)	0.838	0.844	0.841	0.830	0.910	0.869	0.802	0.855	0.827
Parasiam (FastText)	**0.855**	**0.857**	**0.856**	**0.864**	**0.921**	**0.891**	**0.837**	0.859	0.848

Table 1: Performance of our approach and the baseline models. AntSynNet corresponds to the work presented by Nguyen et al. (2017) and Baseline (concat) to a feed forward network on vectors concatenation. The third row refers to the parasiamese model without including the relaxed loss function, and the fourth to the model without performing the pre-training stage. Third and fourth row results were carried out using FastText vectors. Fifth and sixth rows show the results of the complete model (i.e. using pre-training and the relaxed loss function) on ElMo and FastText vectors, respectively.

4.1 Antonymy Dataset

To perform our experiments we use the dataset created by Nguyen et al. (2017). This dataset contains a large amount of pairs of antonyms and synonyms grouped according to its word class (noun, adjective and verb). This dataset was built using pairs extracted by Nguyen et al. (2016) from WordNet and Wordnik[4] to induce patterns through a corpus. Then, the induced patterns were used to extract new pairs, filtering those that match less than five patterns. Finally, the dataset was balanced to contain the same number of antonyms and synonyms, and split into train, validation and test. The number of pairs contained on each partition of each word class is showed in Table 2.

	Train	Val	Test
Adjective	5562	398	1986
Verb	2534	182	908
Noun	2836	206	1020

Table 2: Nguyen et al. (2017) number of word pairs of each partition in the dataset.

4.2 Pre-trained word vectors

For the experimental setting we consider pre-trained general purpose word vectors. We avoid out-of-vocabulary terms using character based approaches. The following publicly available resources were considered:

- FastText (Joulin et al., 2016) vectors trained on English Wikipedia dump [5]. We use default hyper-parameters and vectors dimension is 300.
- ElMo (Peters et al., 2018) vectors for English from Che et al. (2018) [6]. We use the first layer of ElMo that gives representations for decontextualized words.

In the case of FastText we compute 300 dimensional vectors for each word in the dataset. In the case of ElMo embeddings, the pre-trained model was already defined to generate representations of 1024 dimensions.

4.3 Base Network Structure

The base network transforms each word vector into a representative form synonymy and antonymy. Any differentiable function that inputs and outputs vectors of the same dimension of the word embeddings space can be used as base network. In this work we consider layered fully connected networks with ReLU as activation function.

The presented model involves tens of hyper-parameters and some of them with many options. We use random search to find a good hyper-parameter configuration, since it may lead to a better and more efficient solution in comparison to grid or manual search (Bergstra and Bengio, 2012). This improvement is given by the fact that some hyper-parameters do not really matter and grid or manual search would consume time exploring each combination of them (for each combination of the rest), while random search does not exhaustively explore irrelevant parts of the hyper-parameters space.

[4] http://www.wordnik.com
[5] http://mattmahoney.net/dc/enwik9.zip
[6] http://vectors.nlpl.eu/repository/11/144.zip

We perform random search sampling models according to the following considerations:

- 2,3,4 and 5 layers uniformly chosen
- for each hidden layer (if any) we sample its size from a Gaussian distribution with $\mu = d/2$ and $\sigma = d/5$, where d is the dimension of the word vectors.[7]
- dropout with $1/2$ of probability to be activated or not and dropout probability is given by a Gaussian distribution ($\mu = 0.25$ and $\sigma = 0.1$).
- prediction (or positive) threshold and contrastive loss threshold uniformly chosen between $\{2.5, 2, 1.5, 1, 0.5, 0.2\}$ and $\{3, 5, 10\}$, respectively.
- batch size uniformly chosen from $\{32, 64, 128\}$
- we choose between SGD and Adam with equal probability with a learning rate chosen from $\{0.01, 0.001, 0.0001\}$
- the patience for the early stopping was sampled uniformly from $\{3, 4, 7, 9\}$

We initialize the weights of the network using Glorot uniform function (Glorot and Bengio, 2010). We stop the training using early stopping and we checkout the best model in the whole run against the validation set. For the implementation we use Keras (Chollet et al., 2015).

After analyzing the results of 200 sampled hyperparameter configurations using the FastText vectors we found that an adequate hyperparameters setting is a four layered network of input dimensions $[300, 227, 109, 300]$ on its layer from input to output, without dropout, and ReLu activation function for every neuron. For training, a batch size is 64, an acceptance threshold of 2.0 and of 3.0 for the negative part of the contrastive loss. The optimizer method is SGD with a learning rate of 0.01 and a patience of 5 for the early stopping. This training setup was used in both phases: pre-training and training. For the experiments with ElMo embeddings we uniquely adjust hidden layers sizes, probably ElMo results may improve by a dedicated hyperparameter search.

[7]The dimension of input and output layers is d by model definition.

5 Results

In this section we discuss the results obtained with the presented approach and we analyze the model behavior through the outputs of the base network in siamese and parasiamese schemes. We include two baselines with different motivations for comparison purpose. We analyze the model outputs for related and unrelated pairs (i.e. pairs that are not synonyms or antonyms). In the end of this section, we analyze the output of the base network.

5.1 Baselines

We consider two baselines to compare our experiments. The first baseline is a feed forward network classifier that consumes the concatenation of the embeddings of each word in the candidate pair. This baseline compares the performance boost of the proposed model against a conventional supervised classification scheme using neural networks. For this baseline we consider the FastText vectors to feed a four layered network with layer dimensions of $[600, 400, 200, 1]$ from input to output and ReLu as activation function. This model was trained through binary cross-entropy loss and SGD with a learning rate of $0, 01$.

The second baseline we consider for comparison is AntSynNet (Nguyen et al., 2017), a pattern-based approach that encodes the paths connecting the joint occurrences of each candidate pairs using a LSTM. It relies on additional information, such as, part-of-speech, lemma, syntax and dependency trees.

5.2 Antonymy and Synonymy discrimination

We evaluate our model in the antonymy-synonymy discrimination task proposed by Nguyen et al. (2017). However, the task here is faced from a different point of view. In this work we are interested in showing that word vectors contains what is needed to distinguish between antonyms and synonyms, instead of resolving the general task using any available resource. For that reason we do not try to improve the performance adding more information to the model, such as, paths. It is a supervised approach that discriminate antonymy and synonymy using only word vectors as features.

The obtained results are reported in Table 1. The first baseline is included to compare the performance of our model with a word vector concatenation classification. We also report results

with and without pre-training to show the performance gain that pre-training contributes. Notice that, in contrast to AntSynNet, no path-based information is considered in our approach.

5.3 Siamese and parasiamese outputs

In this section we show the outputs of the siamese and parasiamese networks on word pairs chosen from the validation set (see Table 3).

$Word_1$	$Word_2$	Cos	Siam	Psiam
cold	warm	0.327	3.051	1.01
cold	hot	0.367	3.576	0.801
raw	hot	0.467	5.398	1.424
peace	war	0.448	6.167	0.367
stupid	clever	0.406	7.635	1.192
stretch	contract	0.526	4.351	0.354
love	hate	0.378	2.771	0.153
reject	take	0.528	2.341	0.66
close	harsh	0.544	4.11	0.682
small	large	0.178	4.076	0.302
auntie	uncle	0.351	1.721	0.561
day	night	0.366	1.098	0.803
sloping	vertical	0.331	0.254	2.687
hot	cool	0.253	1.753	2.239
invisible	visible	0.137	1.816	2.11
change	mutate	0.593	0.853	4.879
ample	large	0.398	0.115	4.05
flee	depart	0.499	0.682	3.958
cure	heal	0.332	0.646	5.769
elegant	classy	0.48	0.964	5.917
herald	hail	0.507	0.752	5.826
live	exist	0.527	0.126	4.737
agitate	disturb	0.282	0.627	4.683
chop	divide	0.657	1.56	4.085
sturdy	hardy	0.333	1.894	3.493
stout	robust	0.541	1.903	4.361
scatter	dot	0.477	2.05	1.402
fizzle	fail	0.437	3.706	3.464
see	discern	0.501	3.878	3.724

Table 3: A word sampling and their vector cosine distances, siamese and parasiamese (Psiam) outputs. The upper and lower parts correspond to pairs in the validation data as antonyms and synonyms, respectively. The threshold for acceptance is 2.0.

It can be observed in the obtained results, in general, a suitable behavior of the model. We also include the cosine distance to compare and show that it is unable to distinguish between antonyms and synonyms. It is interesting to notice, for instance, in the upper part of the table, that corresponds to antonyms, the difference in outputs between the pairs *cold-warm* and *cold-hot*. It may be interpreted as that *cold-hot* are *more antonyms* than *cold-warm*, which seems adequate. Below the dashed line of each part we include some failure cases.

5.3.1 Non-related pairs

The task setting considered for this work only uses synonyms and antonyms for training. It is interesting to notice that in this work the behavior of the model with unrelated pairs is learned from related pairs and word embeddings, without considering any unrelated pairs during training. We show in Table 4 the outputs given by siamese and parasiamese networks on unrelated pairs.

$Word_1$	$Word_2$	Cos	Siam	Psiam
see	disturb	0.579	3.803	3.866
flee	cure	0.534	3.189	3.764
man	wolf	0.507	4.017	2.649
ascend	speak	0.63	1.927	2.776
safe	adverse	0.475	4.57	0.625
change	mature	0.511	1.118	3.602
cold	night	0.48	1.134	1.98
cold	day	0.574	3.727	0.483
warm	night	0.462	0.773	0.658
warm	day	0.52	0.733	1.601

Table 4: Unrelated pairs and its word vectors cosine distance, siamese model output and parasiamese (Psiam) output.

The obtained results show that the model is not capable to detect unrelated pairs correctly. In fact, the model seems to learn a broader relation. For example, the words safe and adverse are predicted as antonyms and although they are not antonyms, they have some oppositeness. Similarly, the combinations of *cold* and *warm* with *day* and *night* also seems to be coherent since the day tends to be warmer than the night and the night tends to be colder than the day. In the upper part of the table we include unrelated pairs that were correctly predicted as unrelated and below the dashed line we include failure cases on unrelated pairs.

5.3.2 Base Network Output

In this section we analyze the learned base network. In Figure 4 we show a 2D visualization of the original and the transformed word embeddings. The sample of words was chosen from

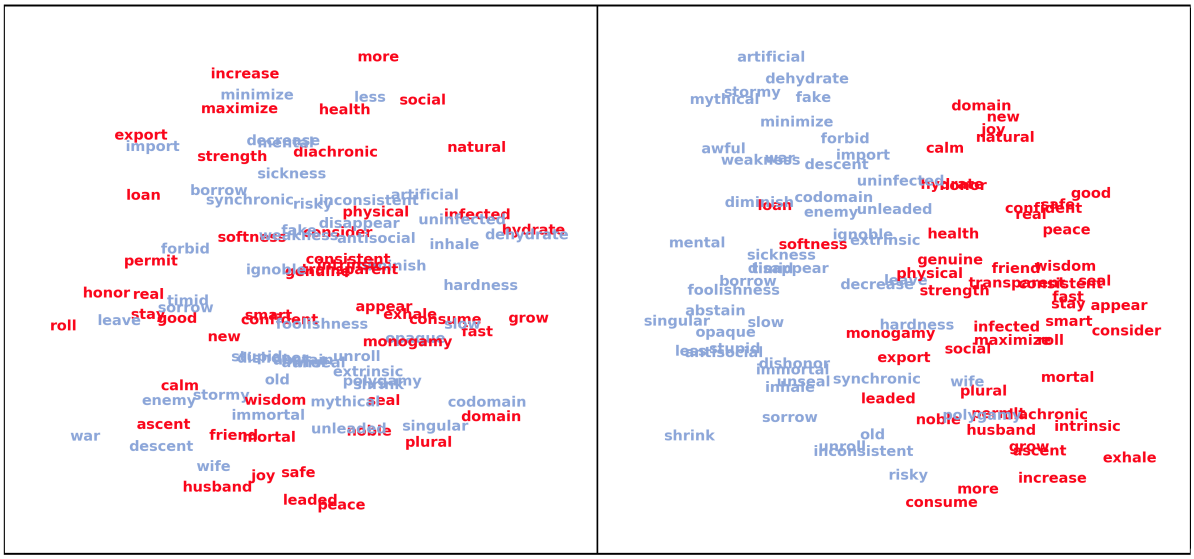

Figure 4: 2D visualization of the original (left) and the transformed (right) words vectors. Related words are colored in red and light blue to facilitate the visualization of how the model split antonymy from original embeddings.

the validation set and t-SNE (Maaten and Hinton, 2008) was used for the dimensionality reduction. It can be observed that in the original space antonyms tend to be close and when the base network is applied the space seems to be split into two parts, corresponding to each pole of antonymy.

We also consider the resulting space from applying the transformation twice to the original word vector space, which is similar to the result of applying it only once. This behavior is coherent with the parasiamese network definition.

To conclude this section, we show the closest words (in the vocabulary) of the words *natural* and *unnatural*, in the original and the transformed spaces, sorted by distance (Table 5). Note how some opposite words appear close in the original space, while in the transformed space the nearest words does not seem to be opposite to the word in question.

word		neighborhood
natural	O	naturals, nonnatural, naturalness, unnatural, naturalmotion, connatural, sobrenatural
	T	pop, morning, simpleness, pee, public, cardia, liveness
unnatural	O	nonnatural, unnaturalness, connatural, unnaturally, naturalness, natural, sobrenatural
	T	shumen, simpsons, untroubledness, hither, random, bewitched, diarrhetic

Table 5: Nearest word vectors in the original (O) and the transformed (T) spaces.

6 Conclusion

We presented a supervised approach to distinguish antonyms and synonyms using pre-trained word embeddings. The proposed method is based on algebraic properties of synonyms and antonyms, principally in the transitivity of synonymy and the antitransitivity of antonymy. We proposed a new siamese inspired model to deal with antitransitivity, the parasiamese network. In addition, we proposed to pre-train this network, relying on the claim that two antonyms of the same word tend to be synonyms, through a siamese network; and a relaxed version of the contrastive loss function. We evaluated our approach using a publicly available dataset and word vectors, obtaining encouraging results.

References

Kfir Bar and Nachum Dershowitz. 2010. Using synonyms for arabic-to-english example-based translation. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas (AMTA 9)*.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305.

Luca Bertinetto, Jack Valmadre, João F. Henriques, Andrea Vedaldi, and Philip H. S. Torr. 2016. Fully-convolutional siamese networks for object tracking. *CoRR*, abs/1606.09549.

Anthony Bonato, Ewa Infeld, Hari Pokhrel, and Paweł Prałat. 2017. Common adversaries form alliances: modelling complex networks via anti-transitivity. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 16–26. Springer.

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium. Association for Computational Linguistics.

Ziming Chi and Bingyan Zhang. 2018. A sentence similarity estimation method based on improved siamese network. *Journal of Intelligent Learning Systems and Applications*, 10(04):121.

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

David Alan Cruse. 1986. *Lexical semantics*. Cambridge university press.

H. P. Edmundson. 1967. Axiomatic characterization of synonymy and antonymy. In *COLING 1967 Volume 1: Conference Internationale Sur Le Traitement Automatique Des Langues*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9 of *JMLR Proceedings*, pages 249–256. JMLR.org.

Aria Haghighi, Andrew Ng, and Christopher Manning. 2005. Robust textual inference via graph matching. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*.

Felix Hill, Kyunghyun Cho, Sébastien Jean, Coline Devin, and Yoshua Bengio. 2014. Embedding word similarity with neural machine translation. *CoRR*, abs/1412.6448.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Chandra Khatri, Gyanit Singh, and Nish Parikh. 2018. Abstractive and extractive text summarization using document context vector and recurrent neural networks. *CoRR*, abs/1807.08000.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

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. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

George A Miller and Walter G Charles. 1991. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28.

Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017. Distinguishing antonyms and synonyms in a pattern-based neural network. *CoRR*, abs/1701.02962.

Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2016. Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction. *arXiv preprint arXiv:1605.07766*.

Masataka Ono, Makoto Miwa, and Yutaka Sasaki. 2015. Word embedding-based antonym detection using thesauri and distributional information. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 984–989.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Herbert Rubenstein and John B Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Magnus Sahlgren. 2008. The distributional hypothesis. *Italian Journal of Disability Studies*, 20:33–53.

Enrico Santus, Qin Lu, Chu-Ren Huang, et al. 2014. Taking antonymy mask off in vector space. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing*.

Silke Scheible, Sabine Schulte Im Walde, and Sylvia Springorum. 2013. Uncovering distributional differences between synonyms and antonyms in a word space model. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 489–497.

Roy Schwartz, Roi Reichart, and Ari Rappoport. 2015. Symmetric pattern based word embeddings for improved word similarity prediction. In *Proceedings of the nineteenth conference on computational natural language learning*, pages 258–267.

Rion Snow, Lucy Vanderwende, and Arul Menezes. 2006. Effectively using syntax for recognizing false entailment. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 33–40. Association for Computational Linguistics.

Ivan Vulić. 2018. Injecting lexical contrast into word vectors by guiding vector space specialisation. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 137–143.

Dexing Zhong, Yuan Yang, and Xuefeng Du. 2018. Palmprint recognition using siamese network. In *Chinese Conference on Biometric Recognition*, pages 48–55. Springer.