Learning Turkish Hypernymy Using Word Embeddings

Savaş Yıldırım [1] **, Tuğba Yıldız** [1]

[1] Department of Computer Engineering, İstanbul Bilgi University
Eski Silahtarağa Elektrik Santralı, Kazım Karabekir Cad. No: 2/13, 34060, Eyüp, İstanbul, Turkey
Tel : +90-212-3117506

E-mail: savas.yildirim@bilgi.edu.tr
E-mail: tdalyan@bilgi.edu.tr

Received 25 January 2017

Accepted 5 December 2017

Abstract

Recently, Neural Network Language Models have been effectively applied to many types of Natural Language Processing (NLP) tasks. One popular type of tasks is the discovery of semantic and syntactic regularities that support the researchers in building a lexicon. Word embedding representations are notably good at discovering such linguistic regularities. We argue that two supervised learning approaches based on word embeddings can be successfully applied to the hypernym problem, namely, utilizing embedding offsets between word pairs and learning semantic projection to link the words. The offset-based model classifies offsets as hypernym or not. The semantic projection approach trains a semantic transformation matrix that ideally maps a hyponym to its hypernym. A semantic projection model can learn a projection matrix provided that there is a sufficient number of training word pairs. However, we argue that such models tend to learn is-a-particular-hypernym relation rather than to generalize is-a relation. The embeddings are trained by applying both the Continuous Bag-of Words and the Skip-Gram training models using a huge corpus in Turkish text. The main contribution of the study is the development of a novel and efficient architecture that is well-suited to applying word embeddings approaches to the Turkish language domain. We report that both the projection and the offset classification models give promising and novel results for the Turkish Language.

Keywords: Word Embeddings, Semantic Relation Projection, Semantic Relation Classification.

1. Introduction

The discovery of semantic relations plays essential role in a variety of Natural Language Processing (NLP) tasks. As one of the important semantic relations, hypernymy indicates is-a relation between two words **a** and **b**. The notation (**a** $\rightarrow$ **b**) indicates that **a** is a hyponym of **b**, and **b** is a hypernym of **a**, e.g. (cat $\rightarrow$ animal).

A considerable amount of studies have been presented on hypernymy so far. Over the last two decades, these studies generally have employed pattern-based and distributional models. In the pattern-based approach, the pioneer study, Ref. [14], proposed a precise acquisition methodology that relies on Lexico-Syntactic Patterns (LSPs) for hypernym relation. However, LSPs often suffer from low recall. Besides, the distributional hypothesis, Ref. [3], assumes that semantically similar words share similar contexts. Some studies have been conducted to detect hypernymy using the distributional hypothesis, Ref. [16,35,44]. For distributional similarity, each word is represented by its contexts in the form of a high-dimensional sparse vector and some similarity functions such as cosine similarity are used for a comparison, Ref. [42].

High-dimensionality and sparsity can be ambiguous and insufficient. Recently neural-network inspired models have been effectively used for vector representation, namely word embeddings, in the form of dense and low-dimensional real-valued vectors. These model have been effectively applied to the semantic and syntactic problems. In the sequel, a word embedding shall be denoted as a mapping $V \rightarrow \Re^D : w \rightarrow \widetilde{w}$ that maps a word w from a vocabulary V to a real-valued vector $\widetilde{w}$ in dimensionality D.

The word embeddings are recently gaining more attention and may help to address a broad range of NLP applications such as multi-task learning (part-of-speech tagging, chunking, named entity tags, semantic role labeling, language model, semantically related words) Ref. [6,7,8,13,41], adjectival scales Ref. [15], text classification Ref. [18], sentiment analysis Ref. [21,38,40], dependency parsing Ref. [1,5], analogies Ref. [11,22,26,30], paraphrase detection Ref. [37], recommendation system Ref. [2] and machine translation Ref. [19].

Many training approaches have been proposed for word embeddings in Ref. [4,7,41]. Recently, Ref. [25] showed that word embeddings are good at capturing syntactic and semantic regularities, using the vector offsets between word pairs sharing a particular relation. For example, if we denote the vector of *word* as x_{word} for comparative/superlative relation, it was found that $x_{big} - x_{bigger} \approx x_{small} - x_{smaller}$, $x_{big} - x_{biggest} \approx x_{small} - x_{smallest}$, and so on. The vector offsets are also used to represent the shared semantic relation between the word pairs. A well-known example is that the offset between the vectors for “queen” and “king” lies very close to the offset between “woman” and “man”, i.e. $x_{queen} - x_{king} \approx x_{woman} - x_{man}$. All these examples show that the word vectors capture the syntactic and semantic regularities. This capacity has been tested against analogy questions that are formulated as in a-b = c-d, where d is unknown and can be estimated by the formula x_d= x_b - x_a + x_c.

Later, Ref. [24] showed how word embeddings are efficiently trained with different architectures, namely the Continuous Bag of Words (CBoW) and the Skip-gram (SG) training model in order to minimize computational complexity and maximize accuracy. These studies have often been an inspiration to other researchers in the field for relation similarity prediction and classification tasks.

In this study, we present and compare two approaches based on word embeddings to solve the hypernym problem in Turkish language, namely, the word embedding offset classification and the semantic projection. While the embedding offset model labels a given word pair as hypernym or not, the semantic projection model maps the embedding of a given word to that of its hypernym. Both approaches are based on supervised learning and require a sufficient number of labeled word pairs. We show that these approaches are successfully and effectively applied to hypernym acquisition for the Turkish language.

2. Related Work

Semantic relation studies include three tasks: relation acquisition, relation classification and relational similarity. The acquisition task aims to automatically extract all possible pairs in a given relation. The classification task includes mapping a word pair to the correct relation from a pre-defined semantic relation set. The relation similarity measures the similarity score between two given word pairs as applied in an analogy task. It checks if a word pair (a,b) stands in the same relation as another word pair (c,d).

Word embeddings approach has became the popular technique to solve these types of tasks because of its good performance. In Ref. [24,25], word embedding has been utilized to capture a considerable amount of syntactic/semantic relations. The vector-offset based methods using simple algebraic operations and cosine similarity have been successfully applied to analogy questions. These important techniques often provided insights about several different studies on relation learning that have been proposed in the literature. Ref. [32] utilized the vector offset technique to complete the task of generating hyponyms. An averaged vector is learned for the hyponym relation by averaging all the offset vectors. Then, the averaged vector is added to a given word to find its hypernym.

Another similar approach was applied to extract hypernym candidates in Ref. [31]. The authors' main assumption was that hypernyms may be induced by adding a vector offset to the corresponding hyponym word embedding. However, they concluded that the diversity involved in the complicated hypernym-hyponym semantic relations cannot be extracted by a simple vector embeddings offset mean. Ref. [33] utilized the vector concatenation and difference on the task of detecting lexical entailment and showed that the study can be used to train a detector to identify several different kinds of Hearst's patterns. Ref. [47] proposed a supervision framework to identify hypernymy relations. The authors designed a dynamic distance-margin model to learn term embeddings and then used the embeddings as input features to further train a supervised classifier to identify hypernymy.

Ref. [10] proposed another embedding-based model to identify hypernym-hyponym relation for Chinese language. In this model, a target word x can be mapped to its hypernyms y based on transition matrix Φ such as $y = \Phi\, x$. The paper proposed two projection approaches: Uniform linear projection and Piecewise linear projection. In Ref. [39], instead of the learning transition matrix Φ, the vector of "is_a" token in the corpus was learned. So, the multiplication of the vector v(w) of any given word w and v(is_a) yields a vector that lies very close to the vector of the hypernym of word w.

Ref. [43] proposed a classification model based on embeddings to learn lexical relations with two different approaches: clustering and classification. The relation types are divided into three categories: lexical semantic, morphosyntactic paradigm and morphosemantic relations. While many morphosyntactic paradigm and morphosemantic relations were captured by vector offsets in clustering task, lexical semantic relations were captured well through classification task. Ref. [27] also proposed another classification model to classify the word pairs into one of three lexical-semantic relations: co-hyponym, hypernym and meronym.

For Turkish language, Ref. [36] trained the word embeddings and measured the performance of the model against manually prepared semantic and syntactic analogy questions which are considered counterpart of English language Ref. [25]. Another word embedding study in Turkish language was done by Ref. [29]. The relations were extracted from the Wikipedia corpus to enrich ontology.

3. Methodology

In this study, we propose two different approaches to solve the problem of hypernym relation in Turkish language; Offset-based binary classification and semantic projection learning. Offset-based classification model takes both negative and positive hypornym-hypernym pairs, while the negative set includes the ones such as <chair, animal>, the positive set includes pairs such as <Denmark, country>. A list of word pairs (w_i,w_j) are encoded into vector embeddings offsets as predictors and their relation are considered to be a binary target class as shown in $(w_i,w_j) \rightarrow \{true, false\}$. The problem is known as statistical binary classification.

The second approach is based on a semantic projection that learns a semantic projection from a list of unambiguous hypernym pairs. Once trained a uniform transition matrix from the hypernym pairs, it simply further maps any given word to its hypernyms. The approaches based on embedding projection have been successfully applied especially in machine translation Ref. [23]. While offset-based classification builds a classifier by taking into account the differences of word embedding vectors, the projection approach learns a uniform transition matrix that maps one vector to another. While the former needs a sufficient number of positive and negative examples, the latter needs only positive pairs to train a model. For classification, the most appropriate models are Support Vector Machine (SVM), Artificial Neural Network (ANN), or logistic regression because these models conform with numeric variables. Semantic projection can be roughly learned using the following Eq. (1), where a target word A can be mapped to its hypernyms B based on a transition matrix Φ such as $B = \Phi\, A$.

$$\Phi^* = argmin_{\Phi}\frac{1}{N}\sum_{(A,B)}||\Phi A - B||^2. \qquad (1)$$

As the formula is applied to training data, minimizing the mean squared error produces a projection matrix. It is simply considered a multiple linear regression, since there are multiple input variables. The most common technique recently employed in machine learning literature is the Gradient Descent Procedure. It consists of a process optimizing the values of the coefficients by iteratively minimizing the error of the linear model on training data. It starts by assigning random values to the model coefficients and repeats the optimization process until no additional improvement is possible or a specific error threshold is achieved. In the semantic projection problem, the output is a vector of size K, rather than a scalar value. To adopt the regression model, we train K linear models rather than a single model that predicts a numeric vector.

One can argue that it is a worthwhile determining whether word embeddings are efficient enough to capture hypernym relations. The question is to determine if such offset-based classifiers or a learned projection matrix can outperform a simple baseline algorithm and the conventional n-gram models. We used the averaging function as baseline algorithm. Some studies Ref. [28,31,32] employ the averaging function to solve the problem. They compute the mean of hypernymy offsets and employ the mean vectors as a stable representation of a particular semantic relation. A given candidate pairs are then evaluated on the cosine similarity closeness to the mean vector. These studies claimed that by adding the averaged offset to a given word one can estimate its hypernym.

3.1. *Preliminary Experiments with Dimensionality Reduction*

To see the applicability of such an approach, we conduct some preliminary experiments. The encouraging observations show that offsets may have significant potential within both the classification and projection models. We compiled a set of positive and negative examples; hypernym (735), non-hypernymy (1647). All pairs were randomly and manually selected. Hypernym relations were derived from 20 unique hypernyms such as animal, country, emotion, etc.

The embeddings have been trained from a large Turkish corpus with size of 1.8G tokens. The t-Distributed Stochastic Neighbor Embedding (t-SNE) approach [a] is applied to understand the hidden patterns. This is a powerful technique for dimensionality reduction, particularly useful for the visualization of high-dimensional data. The projection of offsets along with their relation label is mapped onto a 2D coordinate system as seen in Fig. 1. The figure shows that while hypernym-hyponym relations are properly scattered in the clusters, unrelated relations are disorderly scattered so that they cannot be clustered. One can say that a general model might not fit hypernym-hyponym relations, thus a fine-grained decomposition model could be applicable.

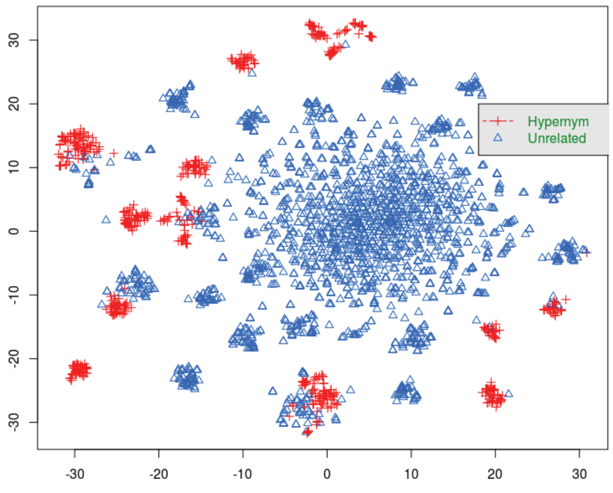

Fig. 1. t-SNE projection of offsets for hypernym and unrelated pairs

Ref. [10] showed similar plotting for the Chinese language and proposed that hypernym-hyponym relations need to be decomposed into more fine-grained sub-relations. Fig. 1 also shows dense clusters associated with certain words such as country, animal, etc. Even though such clusters include pos-

[a] TSNE-R: https://github.com/jdonaldson/rtsne/

itive and negative labels, we also observe that those labels can be separable and a linear model can sufficiently fit the data. Briefly saying, t-SNE projection indicates that the model can distinguish positive and negative examples from each other. However, to induce a general linear model capturing the **is-a** relation is still challenging. The problem requires more fine-grained mapping as can be deduced from our experimental results supporting these preliminary observations.

3.2. Preliminary Experiments with Analogy Questions

We also test our embeddings to answer the analogy questions that are very similar to those designed in English language by Ref. [25]. The study of Ref. [25] comprises a list of syntactic and semantic analogy questions to prove the idea that the word embeddings differences capture syntactic and semantic regularities. As such, the study is able to answer questions with about 40% accuracy in English language. The list contains analogy questions of the form a:b::c:d, where the question is what would the word d be, when a,b and c are given. We find x_a,x_b,x_c embedding vectors and compute y=x_b-x_a+x_c. We search the word whose embedding vector has the greatest cosine similarity to y and computed as in the following Eq. (2)

$$w^* = argmax_w \frac{x_w y}{||x_w||\ ||y||}. \qquad (2)$$

Ref. [36] complied the Turkish counterpart of this analogy question set. It contains around 2000 analogy questions regarding semantic and syntactic regularities. The study applies the trained word embeddings to find the 'word d's in the questions. We also test our trained embeddings against the same questions. We get comparable results with previous Turkish study. Semantic questions are answered with an accuracy of 29% and syntactic ones are answered with an accuracy of 43%.

4. Experimental Setup

In this work, the codes were mostly implemented with the Python programming language and some of its libraries: nltk [b], sklearn[c]. The word embeddings vector was trained by the Python Gensim library [d]. R and Weka platforms were also used for obtaining immediate results, K-fold cross validation, data visualization and some other analyses. All the code implementations and other resources are accessible under the Github platform [e]and the obtained results are reproducible.

Thanks to MTM Media Monitoring Center, [f]a textual corpus were compiled from digital archives of the news agency in the years 2010 to 2014. The corpus are manually categorized in several categories such as politics, economics, magazine etc. by the company. The textual data consists of around 1.8G tokens in total. It is considered adequately representative for deep learning studies in terms of size and coverage. NLP tasks in Turkish mostly require the application of a morphological analysis due to morphologically rich structure of language. In this work, we examine the morphological analysis to see its contribution to the accuracy. We also skip phrase detection phase and carry on with single tokens.

4.1. Classification

For training a model, we randomly prepared a list of negative and positive word pairs of hyponym-hypernym relation. The data set contains 735 positive and 1647 negative pairs. The positive list includes 20 unique hypernyms such as animal, fruits, emotion. Once embeddings are learned from the corpus, each word is associated with an embedding vector of size K (300). Many studies experimentally select the size of word vector, K. To see the impact of vector size, we gradually vary the dimension size K from 100 to 600, in steps of 100. Accordingly, we decided to set the K value to 300. Ref. [19] points

[b] http://www.nltk.org/
[c] http://scikit-learn.org
[d] https://radimrehurek.com/gensim/
[e] https://github.com/savasy/TurkishWordEmbeddings
[f] MTM: http://www.medyatakip.com.tr

that smaller contextual windows generally gets better precision. We experimentally keep the windows size 5.

As described before, the discovery of hypernym relation is clearly considered a binary-class classification wherein the hypothesis is that there is a strong relation between offset vectors and semantic relations. The supervised models can effectively solve semantic relations utilizing the word embedding offset. Independent variables, offsets, are numeric and dependent variables, hypernym relation is binary. For such problem definition, machine learning literature includes many algorithms; linear regression models (Logistic Regression, Stochastic Gradient Descent (SGD) classifier), Support Vector Machines (linear-SVM), k-Nearest Neighbor (KNN) as a lazy learner and Artificial Neural Network (ANN). We choose SVM and ANN algorithms since these algorithms perform better for numeric variables. The kernel function of SVM is a linear kernel and svm type is C-SVC. The loss function is epsilon-insensitive, where the epsilon value is 0.1. ANN employs back-propagation algorithm to determine the gradient and minimize the error. Then the gradient is multiplied by the learning rate to learn the weights on the network. We use only one hidden layer of 151 units, where the number of epoch is 500 and the learning rate is 0.3. The settings are mostly default parameters of the Weka software.

Ref. [24] applied two log-linear embedding models, the Skip-Gram (SG) and Continuous Bag of Words (CBoW) in order to induce word embedding vectors. Such studies argue that SG performs better for semantic relations. Ref. [19] observed that CBoW vectors give higher precision than SG for both German and English. The study suggests that the reason could be that CBoW vectors tend to be slightly more syntactical compared to SG vectors. The syntactical constraint on synonyms, as they appear in similar contexts, have enough influence for CBoW vectors to perform better. We observe that there is no significant differences between the SG and CBoW training algorithms for the classification model. As another factor in the embedding training phase, Negative Sampling (NS) shows slightly better performance than Hierarchical Softmax (HS).

To evaluate the model performance, accuracy could be misleading metric when using an imbalanced dataset. Imbalanced class distribution is very predominant problem in many fields such as the Non-Technical Losses (NTL), diagnosis of rare diseases, bank fraud, earthquake detection etc. The ratio between the majority and minority classes is even higher than 1000/1. Ref [12] addressed the imbalanced dataset issue by giving a gentle example for NTL problem. There is 1 NTL as positive class against 999 non-NTL as negative class. The study finally underlines that accuracy might lead to highly artificial scores on such imbalanced data sets. Ref [9] also addressed the fact that the Receiver Operating Characteristics (ROC) curve has become increasingly important in the presence of imbalanced classes. The study stated that "true positive fraction" and "false positive fraction" are more meaningful than accuracy. The Area Under ROC Curve (AUC) metric measures how well the classifier separates two classes since it is not sensitive to class distribution. On the other hand, ref.[34] argued that Precision-Recall Plot (PRC) could be more informative than AUC under some circumstances.

Beside, during training phase there are some approaches to handling imbalanced datasets. Resampling training data is a very common way for the problem where we can replicate the instances of under-represented *(minority class)*, called over-sampling, or discard the instances of over-represented *(majority class)* to balance the class distribution, called under-sampling. An alternative to the first approach is synthetically sampling minority examples other than replicating to reach the uniform class distribution, called SMOTE, ref [48]. It randomly selects an instance and produces similar synthetic instances by using the distances to its neighboring instances. We applied these resampling techniques to our problem. Although the techniques slightly improved F1 score of the machine learning algorithms, they did not show any performance differences in terms of AUC metrics.

As shown in Table 1, the ANN algorithm outperforms other algorithms with the best F1, AUC and PRC scores, 88.3% 97.5% and 92.8% respectively. SVM with linear kernel has relatively smaller scores, 81.1% F1, 87.2% AUC and 70.7% PRC. Some offset-based studies use averaging model for classification. Instead, we employ it as a baseline algorithm. Baseline algorithm using averaging model gets low rates such as 79.8% F1, 84.1% AUC and 70.3% PRC.

The results suggests that ANN and SVM classifiers notably outperform the baseline algorithm. The Kappa index is also helpful to check reliability of such scores, where the Kappa value measures how much better, or worse, a classifier is compared with a random model. Since using accuracy only might be misleading, Kappa Statistic compares the accuracy of any system to the accuracy of a random system. It compares an observed accuracy with an expected accuracy based on random chance. The Kappa values of three algorithm are 83.0%, 73.0% and 70.0% in order. As a final remark, even though the ANN has a good capacity as a classifier, the main deficiency of ANN is its undesirable running time complexity.

Table 1. Performances of the Classification Algorithms: The scores are validated by 10-fold cross validation. (SR: Success Rate, KI: Kappa Index, F1: F-Measure, P: Precision, R: Recall, AUC: Area Under ROC, PRC: Precision-Recall AUC)

Model (%)	SR	KI	F1	P	R	AUC	PRC
ANN	93.2	83.0	88.3	86.3	90.4	97.5	92.8
SVM	89	73.0	81.1	79.7	82.6	87.2	70.7
AVG.*	88.0	70.0	79.8	77.5	80.1	84.1	70.3

4.1.1. Challenges in Classification

Ref. [44] addressed a significant issue while evaluating their experiments. They employ the additional constraint of prohibiting shared words between training and testing data. They conclude that offsets-based learning methods can achieve artificially high scores as a consequence of lexical memorization. The classifiers generally associate frequent words with the hypernym relation. For example, the terms such as bird, cat, snake all have the superclass animal, and the model further classifies any word pair including the frequent term animal as a positive example. In other words, they claimed that word embedding vectors encode some notion of generality which lets classifiers decide that a particular frequent word is likely to have a hypernym relation, irrespective of accompanying word.

We address this memorization problem by proposing another scenario. We prepared a particular dataset that has a list of positive and negative examples of a certain frequent hypernym. This experiment is individually conducted for four different frequent words; country, animal, occupation and fruit. As shown in Table 2, each test implies that offset based classifiers, where ANN is used, successfully separate the instances as positive and negative, contrary to lexical memorization expecting that each instance is labeled positive. Same property has been observed to hold via t-SNE projection of the offsets. The positive and negative offsets distribute such a way that the placements are linearly separable. We clearly observe that a linear model might be a good separation. Consequently, the supervised model is able to take the second term into account to make a decision. However, there still exists an another issue, namely, if the training data lacks instances of a particular hypernym, the models hardly find the hypernym-hyponym relation corresponding to the particular missing hypernym. Training algorithm does not offer a generalized model for those instances. We conducted an experiment to address

Table 2. Performance of ANN algorithm against Lexical Memorization problem: Instances of a certain hypernym are taken for training set. (SR: Success Rate, KI: Kappa Index, F1: F-Measure, P: Precision, R: Recall, AUC: Area Under ROC, PRC: Precision-Recall AUC)

Hypernym (%)	SC	KI	F1	P	R	AUC	PRC
Country	96.9	92.3	98.0	98.0	98.0	99.2	99.7
Animal	94.6	90.1	94.5	93.1	96.1	96.1	92.8
Occupation	95.5	89.9	97.6	96.5	98.8	97.9	99.0
Fruit	96.8	93.2	94.8	93.7	96.2	99.5	99.5

Table 3. Performance of ANN algorithm by leaving instance of a certain hypernym out from training. (SR: Success Rate, KI: Kappa Index, F1: F-Measure, P: Precision, R: Recall, AUC: Area Under ROC, PRC: Precision-Recall AUC)

Hypernym	SC	KI	F1	P	R	AUC	PRC
Country	46.9	0.14	44.1	93.2	28.9	77.3	88.3
Animal	41.0	-0.002	2.3	25	1.2	54.8	57.3
Occup.	33.2	-0.34	46.2	55.0	39.8	20.7	56.5
Fruit	54.2	0	0	0	0	64.1	53.6

the issue, as shown in Table 3. When leaving all instance of a particular hypernym out from training data and using them as test set, the models get very poor performance in terms of all measures such as AUC, PRC, F-measure.

4.1.2. Other Factors

We investigated the impact of embeddings size, amount of training corpus and morphological analysis. As expected, we observed that system performance is definitely dependent on the vector size up to an optimum point. We checked the effect of the dimension by gradually changing the size from 100 to 600. The optimum pick point seems to be 300. Ref. [19] varied the window size, and the size of dimensions. They observed that the performances of models using the CBoW are better than SG. They also stated that the number of dimensions did not seem to play a very important role. They obtained the best results for dimensions of 300 and 600. They found that the optimal contextual windows size was about 4 for English and 8 for German. To see optimum corpus size, we partially use corpus ranging from 20% to 100%. The study shows that as the training corpus size increases, the model scores rise slightly.

Morphological analysis is an essential phase for many NLP problems especially in morphologically rich language such as Turkish. On the other hand, morphological analysis incurs a large running time complexity. The studies regarding Big Data problems require less complex models and skip some phases due to constraints such as resource allocation. For instance, a real-time social media analytics handle huge data feeds instantaneously. In order to see the contribution of a parser in terms of classifier accuracy, we remove the suffixes and take only stem of the tokens. We observe that there is no improvement but even slight deterioration. We argue that if the corpus size is sufficiently big, no morphological analysis is needed.

4.2. Projection learning

The semantic projection matrix can be learned from a sufficient list of unambiguous hypernym-hyponym word pairs. A projection matrix further maps a given word embedding to an output embedding vector depending on the learned task. The nearest word to the output vector is proposed as a solution. Some studies evaluate their model by taking first K nearest words to the output vector into account. We only use the first nearest word, K=1. This simple approach has been applied to various problems, such as meronym identification, machine translation, corpus alignment Ref. [17,19,23,45].

We take the same hypernym set already used in the classification model discussed in the previous section. There exist 735 unambiguous word pairs in which 20 unique hypernyms exist. The test phase employs 10-fold cross validation. Table 4 shows the performance of the projection model with respect to three loss functions. The loss function "epsilon intensive" is the most suitable one with the accuracy of 88.9%. The other two loss functions, squared loss and squared epsilon, achieve 85.2% and 87.6% respectively.

Table 4. Accuracy of the Semantic Projection Model across three loss function

Loss Function	Accuracy
SQ Loss	85.2
Epsilon_Insensitive	**88.9**
Sq Epsilon_Insensitive	87.6

Even though the obtained score is very promising, the essential deficiency of such a projection learning model is that all the target words (hypernyms) in training data are dominating the projections so that any given word is overwhelmingly mapped to one of those hypernym words in training data. The model hardly maps any given word (hyponym) to its hypernym that does not exist in the training set. It works similarly to multi-class classification model. It has the difficulties in learning a is-a relation just as in the classification model. It furthermore requires at least some seed instances. One can argue that these embedding-based approaches can learn is-a-particular-hypernym relation rather than is-a relation.

To see this effect, we run some additional experiments using a method that leaves specific hypernyms out from training as in the classification model. In each round, a particular hypernym instances extracted from training data are used as test set. So the model does not learn any pattern regarding that hypernym. We applied this leave-hypernym-out validation method with some frequent words as shown in the Table 5. Each row is associated with a particular hypernym that is left out from the training set and shows how the corresponding real hyponyms incorrectly distribute under other hypernyms. Those columns whose column sum is zero are dropped to shrink table, such as sport, drink. We see that under such condition the model cannot map the hyponym to its real hypernym but to the most related ones. E.g. the city instances are never mapped to the hypernym "city" but "country" 79 times, 91%. Likewise, the fruits are never mapped to "fruit" but to "vegetable" 28 times, 65%. That is, the projection model tends to learn, to some extent, the notion of relatedness.

However, one of the robust characteristics of the semantic projection learning model is that even if a few instances of a hypernym is provided in training phase, the model successfully learns the pattern. This might give very high recall scores. We conducted some experiments in order to analyze this property. We built training data by selecting K instances for each hypernym, where we vary K from 2 to 24. The model trained even with 2 instances got a fairly remarkable result with the success score 68%. Fig. 2 shows that as the seed size increases, the model performance naturally gets higher results. Using more than 8 seeds lead to efficient results. The optimum choice is around 16-18.

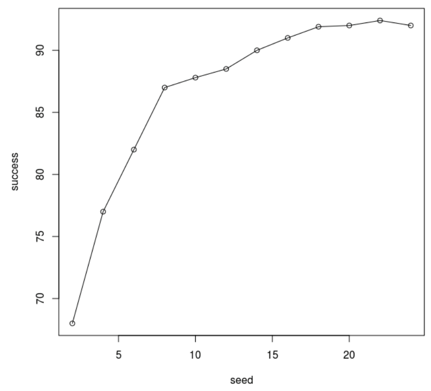

Fig. 2. The performance of semantic projection by the seed size

4.3. Final Remarks and Discussion

We employ the offset averaging method as the baseline algorithm. The classification model gets a successful score of 88.3% F1, 97.5% AUC, 92.8% PRC. It achieved 8.5% F1, 13.4% AUC and 22.5% PRC improvement over the baseline method by 10-fold cross validation. The Kappa values of all the classifiers are at very sufficient level. As a second approach, the projection learning model is considered as hypernym generation model. Table 4 shows that the model using the loss function "epsilon insensitive" gets an accuracy of 88.9%, which is slightly higher than that for the classification model. It also outperforms the averaging model. One important characteristic of projection learning is that training even with few instances under each hypernym leads to a successful classifier. We tests by varying the seed size from 2 to 20. When using eight and more seed instances, the projection model shows promising performance.

The main deficiency of the classification approach is that it poorly decides the instances of a particular hypernym if the training set does not include its instances. It requires at least some seed instances. Interestingly, the same property is observed for the projection learning approach as well. Likewise, lack of hypernym instances in training set leads to poor performance for the projection learning model. For a given hyponym, it mistakenly proposes the most related hypernym instead of the correct one. So, the hypernym list of training set dominates the model such that hyponyms are consistently mapped to one hypernym of the list.

Ref. [44] and Ref. [20] assert that lexical memorization is an important issue because frequent terms always tend to be labeled hypernym by classifiers irrespective of accompanying words. They underline the fact that the success ratio could be artificially high. We checked the issue by conducting many experiments as mentioned above. Our training data had the frequent terms such country, animal along with their positive and negative examples. When we ran the classification model, it successfully separated the classes. We also designed another scenario so that many positive and negative instances of a particular frequent term are taken account as

Table 5. Leaving a certain hypernym out: Each row associated with a particular hypernym depicts that corresponding hypernym is left out from training set and how mappings of its real hyponyms distribute

		Predicted						
		city	vegetable	animal	fruit	occupation	country	Total
	city			1		6	79	86
	sport	1		5			39	45
	vegetable			13	28		2	43
	drink		17	8	1		1	27
Actual	animal	6	3		37	16	21	83
	fruit	1	17	25			1	44
	occupation	13		26	1		43	83
	country	66	9	42		25		142
	news	3	1	3		8	13	28
	Total	90	30	113	38	58	87	

training and test data. As shown in Table 2, the terms country, fruit, animals, occupation were successfully classified without lexical memorization. t-SNE mapping of the offsets also shows these same characteristics.

The architecture proposed in this study exhibits a semi-automatic structure for the problem of hypernymy extraction. To design a fully automatic one, we can utilize additional methods. The classification model needs a list of positive and negative examples for hypernyms. The semantic projection model needs only positive examples. To designate a fully automatic architecture, we envision the utilisation of Lexico-Syntactic Patterns (LSPs) since they have very high precision to produce a positive instance for a given hypernym. In Ref. [46], a fully automatic system for acquisition of hypernym/hyponymy relations was proposed for Turkish Language. The method relies on both LSPs and semantic similarity. The model first utilized LSPs to extract seeds, then it applied n-gram similarity based expansion to increase recall. Likewise, we can design an automated architecture so that it takes initial pairs from LSP patterns and expand the list by applying word embeddings model as discussed in the paper.

5. Conclusion

In this study, we propose two word-embedding based approaches to solve the hypernym detection problem in Turkish: Embedding Offset Classification and Semantic Projection. While the former approach exploits the embedding differences of word pairs and classifies them as hypernym or not, the latter maps a given word to its hypernym by a learned transition matrix. Our experiments suggest that the classification and the projection models show very similar performances. They both achieve a substantial improvement over the baseline algorithm that uses the averaging method. We also address some challenges. If a training set lacks an instance of a certain hypernym, it becomes difficult for both approaches to decide on any instance of that hypermym.

We show that such architectures tend to learn is-a-particular-hypermym rather than is-a generalization. On the other hand, if the training set includes even few instances of hypernym, these architectures successfully achieve the task as the experiments indicate. We also check the lexical memorization issue and report that the models do not show such a characteristic and the achieved scores in the paper are not artificially high. As a future work, syntactical rules are also worth applying to word embeddings especially for the Turkish language as it has morphologically rich characteristics. We hope that both approaches discussed here can be effectively applied to syntactics as well.

Acknowledgments

We are grateful to the news agency, MTM Media Monitoring Center for providing textual data for this study.

References

1. M. Bansal, K. Gimpel, K. Livescu, “Tailoring continuous word representations for dependency parsing,” *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, Maryland, 2014, pp. 809–815.
2. O. Barkan, N. Koenigstein, “Item2vec: Neural item embedding for collaborative filtering,” *CoRR* **abs/1603.04259** (2016).
3. Z.S. Harris, “Distributional structure,” *Word* **10** (1954) 146–162.
4. Y. Bengio, R. Ducharme, P. Vincent, C. Janvin, “A neural probabilistic language model,” *J. Mach. Learn. Res.* **3** (2003) 1137–1155.
5. W. Chen, Y. Zhang, M. Zhang, “Feature embedding for dependency parsing,” *In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, Dublin, Ireland, 2014, pp. 816–826.
6. R. Collobert, J. Weston, “A unified architecture for natural language processing: Deep neural networks with multitask learning”, *Proceedings of the 25th International Conference on Machine Learning*, New York, NY, USA, 2008, pp. 160–167.
7. R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P.P. Kuksa, “Natural language processing (almost) from scratch,” *Journal of Machine Learning Research* **12** (2011) 2493–2537.
8. E. Okur, H. Demir , A. Ozgur, “Named entity recognition on twitter for turkish using semi-supervised learning with word embeddings,” *Proceedings of the 10th Language Resources and Evaluation Conference, LREC2016*, 2016, pp. 549–555.
9. T. Fawcett, “ROC graphs: Notes and practical considerations for researchers,” *ReCALL 31 HP Laboratories*, Palo Alto, 2004, pp. 1–38.
10. R. Fu, J. Guo, B. Qin, W. Che, H. Wang, T. Liu, “Learning semantic hierarchies via word embeddings,” *In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, Maryland, 2014, pp. 1199–1209.
11. J. Gao, P. Pantel, M. Gamon, X. He, L. Deng, “Modeling interestingness with deep neural networks,” *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, 2014, pp. 2–13.
12. P. Glauner, J. Meira, P. Valtchev, R. State and F. Bettinger, “The Challenge of Non-Technical Loss Detection using Artificial Intelligence: A Survey”, *International Journal of Computational Intelligence Systems (IJCIS)* **10**(1) (2017) 760–775.
13. J. Guo, W. Che, H. Wang, T. Liu, “Revisiting embedding features for simple semi-supervised learning,” *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, 2014, pp. pp. 110–120.
14. M. Hearst, “Automatic acquisition of hyponyms from large text corpora”, *Proceedings of the 14th International Conference on Computational Linguistics*, Nantes, France, 1992, pp. 539–545.
15. J.K. Kim, M.C. Marneffe, “Deriving adjectival scales from continuous space word representations,” *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, USA, 2013, pp. 1625–1630.
16. L. Kotlerman, I. Dagan, I. Szpektor,M. Zhitomirsky-geffet, “Directional distributional similarity for lexical inference,” *Nat. Lang. Eng.* **16**(4) (2010) 359–389.
17. V. Kulkarni, R. Al-Rfou, B. Perozzi, S. Skiena, “Statistically significant detection of linguistic change”, *CoRR* **abs/1411.3315** (2014)
18. Q.V. Le, T. Mikolov, “Distributed representations of sentences and documents,” *Proceedings of The 31st International Conference on Machine Learning*, Beijing, China, 2014, pp. 1188–1196.
19. A. Leeuwenberg, M. Vela, J. Dehdari, J. van Genabith, “A minimally supervised approach for synonym extraction with word embeddings,” *The Prague Bulletin of Mathematical Linguistics* **105** (2016) 111–142.
20. O. Levy, S. Remus, C. Biemann, I. Dagan, “Do supervised distributional methods really learn lexical inference relations?,” *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Denver, Colorado, USA, 2015, pp. 970–976.
21. A.L. Maas, R.E. Daly, P.T. Pham, D. Huang, A.Y. Ng, C. Potts, “Learning word vectors for sentiment analysis,” *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Stroudsburg, PA, USA, 2011, pp. 142–150.
22. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. *CoRR* **abs/1301.3781** (2013)
23. T. Mikolov, Q.V. Le, I. Sutskever, “Exploiting similar-

ities among languages for machine translation" *CoRR* **abs/1309.4168** (2013).

24. T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems*, Lake Tahoe, Nevada, United States, 2013, pp. 3111–3119.
25. T. Mikolov, W. Yih, G. Zweig, "Linguistic regularities in continuous space word representations," *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics*, Atlanta, Georgia, USA, 2013, pp. 746–751.
26. A. Mnih, K. Kavukcuoglu, "Learning word embeddings efficiently with noise-contrastive estimation," *Advances in Neural Information Processing Systems 26*, eds. L. Burges, M. Bottou, Z. Welling, K. Ghahramani, K. Weinberger, 2013, pp. 2265–2273.
27. S. Necsulescu, S. Mendes, D. Jurgens, N. Bel, R. Navigli, "Reading between the lines: Overcoming data sparsity for accurate classification of lexical relationships," *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, Denver, Colorado, 2015, pp. 182–192.
28. M. Ono, M. Miwa, Y. Sasaki, "Word embedding-based antonym detection using thesauri and distributional information", *Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL HLT 2015*, Denver, Colorado, USA, 2015, pp. 984–989.
29. İ. Pembeci, "Using word embeddings for ontology enrichment," *International Journal of Intelligent Systems and Applications in Engineering* **4**(3) (2016) 49–56.
30. J. Pennington, R. Socher, C. Manning, "Glove: Global vectors for word representation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, 2014, pp. 1532–1543.
31. J. Pocostales, "Nuig-unlp at semeval-2016 task 13: A simple word embedding-based approach for taxonomy extraction," *Proceedings of SemEval-2016*, San Diego, California, 2016, pp. 1303–1309.
32. M. Rei, T. Briscoe, "Looking for hyponyms in vector space," *Proceedings of the Eighteenth Conference on Computational Language Learning*, Baltimore, Maryland USA, 2014, pp. 68–77.
33. S. Roller, K. Erk, "Relations such as hypernymy: Identifying and exploiting hearst patterns in distributional vectors for lexical entailment," *CoRR* **abs/1605.05433** (2016).
34. T. Saito, M. Rehmsmeier, "The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets". *PLoS ONE* **10**(3) (2015) DOI: 10.3410/f.725375858.793530299.
35. E. Santus, A. Lenci, Q Lu, S.S. im Walde, "Chasing hypernyms in vector spaces with entropy," *roceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, Gothenburg, Sweden, 2014, pp. 38–34.
36. M.U. Sen, H. Erdogan, "Learning word representations for Turkish," *22nd Signal Processing and Communications Applications Conference (SIU)*, Trabzon, Turkey, 2014, pp. 1742–1745.
37. R. Socher, E.H. Huang, J. Pennington, A.Y. Ng, C.D. Manning, "Dynamic pooling and unfolding recursive autoencoders for paraphrase detection," *Advances in Neural Information Processing Systems*, 2011, pp. 801–809.
38. R. Socher, J. Pennington, E.H. Huang, A.Y. Ng, C.D. Manning, "Semi-supervised recursive autoencoders for predicting sentiment distributions," *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Stroudsburg, PA, USA, 2011, pp. 151–161.
39. L. Tan, R. Gupta, J. van Genabith, "Usaar-wlv: Hypernym generation with deep neural nets", *Proceedings of the 9th International Workshop on Semantic Evaluation*, Denver, Colorado, USA, 2015, pp. 932–937.
40. D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, B. Qin, "Learning sentiment-specific word embedding for twitter sentiment classification," *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, Maryland, 2014, pp. 1555–1565.
41. J. Turian, L. Ratinov, Y. Bengio, "Word representations: A simple and general method for semi-supervised learning," *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, Stroudsburg, PA, USA, 2010, pp. 384–394.
42. P. Turney, P. Pantel, "From frequency to meaning: Vector space models of semantics," *Journal Artifical Intelligence Research* **37**(1) (2010) 141–188.
43. E. Vylomova, L. Rimell, T. Cohn, T. Baldwin, "Take and took, gaggle and goose, book and read: Evaluating the utility of vector differences for lexical relation learning," *CoRR* **abs/1509.01692** (2015).
44. J. Weeds, D. Clarke, J. Reffin, D.J. Weir, B. Keller, "Learning to distinguish hypernyms and co-hyponyms," *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*, Dublin, Ireland, 2014, pp. 2249–2259.
45. C. Xing, D. Wang, C. Liu, Y. Lin, "Normalized word embedding and orthogonal transform for bilingual word translation," *Human Language Technologies:*

The 2015 Annual Conference of the North American Chapter of the ACL, Denver, Colorado, USA, 2015, pp. 1006–1011.

46. S. Yildirim, T. Yildiz, "Automatic extraction of turkish hypernym-hyponym pairs from large corpus," *24th International Conference on Computational Linguistics, Proceedings of the Conference*, Mumbai, India, 2012, pp. 493–500.
47. Z. Yu, H. Wang, X. Lin, M. Wang, "Learning term embeddings for hypernymy identification," *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, (2015) 1390–1397.
48. Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. J. Artif. Int. Res. 16, 1 (June 2002), 321-357.