Post-hoc Manipulations of Vector Space Models with Application to Semantic Role Labeling

Jenna Kanerva and **Filip Ginter**
Department of Information Technology
University of Turku, Finland
jmnybl@utu.fi, figint@utu.fi

Abstract

In this paper, we introduce several vector space manipulation methods that are applied to trained vector space models in a post-hoc fashion, and present an application of these techniques in semantic role labeling for Finnish and English. Specifically, we show that the vectors can be circularly shifted to encode syntactic information and subsequently averaged to produce representations of predicate senses and arguments. Further, we show that it is possible to effectively learn a linear transformation between the vector representations of predicates and their arguments, within the same vector space.

1 Introduction

Recently, there has been much progress in the development of highly scalable methods for inducing vector space representations of language. In particular, the *word2vec* method (Mikolov et al., 2013b) is capable of training on billions of tokens in a matter of hours, producing high quality representations. An exciting property exhibited by the vector spaces induced using *word2vec* is that they preserve a number of linguistic regularities, lending themselves to simple algebraic operations with the vectors (Mikolov et al., 2013c) and linear mapping between different spaces (Mikolov et al., 2013a). These can be seen as *post-hoc* operations manipulating the vector space with the significant advantage of not requiring a new task-specific representation to be induced, as is customary.

In this paper, we will investigate several additional such methods. Firstly, we will show how syntax information can be encoded by the circular shift operation and demonstrate that such shifted vectors can be averaged in a meaningful manner to represent predicate arguments. And secondly, we will show that linear transformations of the vector spaces can be successfully applied also within a single vector space, to tasks such as transforming the vector of a predicate into the vector of its argument with a particular role.

To test the above-mentioned operations in an extrinsic setting, we will develop these methods within the context of the Semantic Role Labeling (SRL) task. Automatic Semantic Role Labeling is the process of identifying the semantic arguments of predicates, and assigning them labels describing their roles. A predicate and its arguments form a predicate-argument structure, which describes events such as *who* does *what* to *whom*, *when* and *where*.

The SRL task is "semantic" in its nature and therefore suitable for the application and testing of vector space representations and methods for their manipulation. However, rather than merely adding features derived from vector spaces into an existing system, we will approach the development from a different angle and test whether these representations of words and the similarities they induce can be used for predicate argument role assignment and predicate sense disambiguation as the primary source of information, with little additional features.

In addition to the standard English CoNLL'09 dataset, we will apply the methods also to Finnish SRL, testing the applicability of *word2vec* and the overall methodology that we will develop in this paper to this highly inflective language. With its considerably larger and sparser surface form lexicon, Finnish poses interesting challenges of its own, and only little attention has been dedicated to the application of distributional semantics methods specifically to Finnish. This is also partly due to the lack of sufficiently sized corpora, which we address in this work by using a 1.5B token corpus of Internet Finnish.

In order to be able to test the proposed meth-

Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC) @ EACL 2014, pages 1–10,
Gothenburg, Sweden, April 26-30 2014.

ods on SRL, we need to carry out not only role labeling and predicate sense disambiguation, but also argument detection. As a secondary theme, we thus test whether dependency parse graphs in the semantically motivated Stanford Dependencies (SD) scheme can be used as-is to perform argument identification. We are especially interested in this scheme as it is designed to capture *semantically contentful* relations (de Marneffe and Manning, 2008) and would thus appear to be the ideal choice as the underlying syntactic representation for SRL.

2 Data and Task Setting

Throughout the paper, we will use the exact same task setting as in the CoNLL'09 Shared Task on Syntactic and Semantic Dependencies in Multiple Languages (Hajič et al., 2009). The input of the SRL system are automatically generated syntactic parses and the list of predicate tokens to be considered in each sentence. For each of the predicates, the SRL system is expected to predict the *sense* of the predicate, identify all tokens which are its arguments, and for each argument, identify its role. As the primary measure of performance, we will use the *semantic F-score* defined in the CoNLL shared task. This F-score is calculated from the precision and recall of argument identification (calculated in the obvious manner) and also incorporates the sense of the predicate via an additional "dummy" argument. We use the official implementation of the metric distributed on the Shared Task site.[1]

We will report our results on two SRL datasets: the *Finnish PropBank* (Haverinen et al., 2013a) and the English SRL dataset from the CoNLL'09 Shared Task. The Finnish PropBank is built on top of the *Turku Dependency Treebank (TDT)*, a 205K token corpus of general Finnish (Haverinen et al., 2013b) annotated using the SD scheme, including manually annotated conjunct propagation and other dependency relations from the non-basic layer of the scheme. These extended SD analyzes are thus not strictly trees, rather they are directed labeled graphs (see Figure 1). The Finnish PropBank has 22 different argument roles of which 7 are numbered core roles and 15 are modifier roles. The Finnish data has 164,530 training tokens with 27,603 occurrences of 2,826 unique predicate-sense combinations. The English CoNLL data is derived from the PropBank and NomBank corpora (Palmer et al., 2005; Meyers et al., 2004) and it has a total of 54 different argument roles. In addition to the same 22 roles as Finnish, English also has discontinuous variants for each role. The English data has 958,024 training tokens with 178,988 occurrences of 15,880 unique predicate-sense combinations.

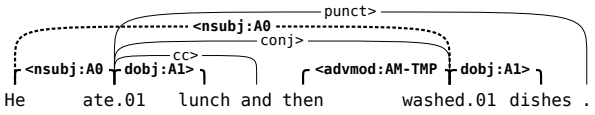

Figure 1: Extended Stanford Dependencies scheme combined with PropBank annotation.

All Finnish results are reported on the test subset of the Finnish PropBank, and have no previously published baseline to compare with. The results we report for English are produced on the official test section of the CoNLL'09 data and are thus directly comparable to the official results reported in the Shared Task.

In the test phase, we follow the Shared Task setting whereby morphological and syntactic analysis is predicted as well, i.e., no gold standard data enters the system other than the tokenization and the information of which tokens constitute predicates. We produce the Finnish morphological and syntactic analyses for the test set with the parsing pipeline of Haverinen et al. (2013b), composed of a morphological analyzer and tagger (Halácsy et al., 2007; Pirinen, 2008; Lindén et al., 2009), dependency parser (Bohnet, 2010) and a machine-learning based component for predicting the extended SD dependencies (Nyblom et al., 2013). While the English data is provided with automatically produced dependency parses, we are specifically interested in the SD scheme and therefore we re-parse the corpus with the Stanford parser[2] taking a union of the base and collapsed dependency outputs to match the Finnish data.

The vector space models used throughout this paper are induced using the *word2vec* software (skip-gram architecture with default parameters). For Finnish, the model is trained on 1.5 billion tokens of Finnish Internet texts gathered from the *Common Crawl* dataset.[3] The data was sentence-

[1] http://ufal.mff.cuni.cz/conll2009-st/scorer.html

[2] Version 3.3.1, October 2013

[3] http://commoncrawl.org/

split and tokenized using the OpenNLP[4] toolchain trained on TDT, and processed in the same manner as the above-mentioned test set. This gives us the opportunity to build two models, one for the word forms and the other for the lemmas. Both Finnish models have 300 dimensions. For English, the vector representation is induced on the union of the English Wikipedia (1.7B tokens) and the English Gigaword corpus (4B tokens), the total training data size thus being 5.7 billion tokens.[5] Sentence splitting and tokenization was carried out using the relevant modules from the *BRAT* package (Stenetorp et al., 2012).[6] The English model has 200 dimensions.

3 Method

In this section, we will describe the methods developed for argument identification, argument role labeling and predicate sense disambiguation, the three steps that must be implemented to obtain a full SRL system.

3.1 Argument identification

In a semantically-oriented dependency scheme, such as SD, it can be expected that a notable proportion of arguments (in the SRL sense) are directly attached to the predicate, and argument identification can be reduced to assuming that — with a limited number of systematic exceptions — every argument is a dependent of the predicate. The most frequent case where the assumption does not hold in Finnish are the copula verbs, which are not analyzed as heads in the SD scheme. For English, a common case are the auxiliaries, which govern the main verb in the CoNLL data and are thus marked as arguments for other higher-level predicates as well. In the SD scheme, on the other hand, the main verb governs the auxiliary taking also its place in the syntactic tree. Since the focus of this paper lies in role assignment, we do not go beyond developing a simple rule set to deal with a limited number of such cases. In Section 6, we will contrast this simple argument identification method to that of the winning CoNLL'09 system and we will show that while for Finnish the above holds surprisingly well, the performance on the English data is clearly sub-optimal.

[4] http://opennlp.apache.org/

[5] We are thankful to Sampo Pyysalo for providing us with the English *word2vec* model.

[6] http://brat.nlplab.org

Finnish	
eat + A1	AM-TMP
salted fish	not until
eggs	now
wheat bread	again
nuts	when
pickled cucumbers	then
English	
drive + A1	drive + AM-TMP
car	immediately
truck	morning
cars	now
vehicle	afternoon
tires	finally

Table 1: Five most similar words for the given average argument vectors. *AM-TMP* refers to the temporal modifier role. Note that the average vectors for Finnish modifier roles are estimated independently from the predicates (see Section 3.4).

3.2 Role Classification

Our initial role classification algorithm is based on calculating the vector representation of an "average argument" with a given role. For every predicate x and every role r, we calculate the representation of the average argument with the role as

$$A(x,r) = \frac{\sum_{(r,x,y)} \hat{y}}{count}\,, \qquad (1)$$

where $\hat{y}$ refers to the L2 normalized version of y, and *count* to the number of training pairs that are summed over. We are thus averaging the normalized vectors of all words y seen in the training data as an argument of the predicate x with the role r. To establish the role for some argument y during testing, we can simply choose the role whose average argument vector has the maximal similarity to y, i.e.

$$\arg\max_r sim(A(x,r), y), \qquad (2)$$

where $sim(a,b)$ is the standard cosine similarity.

To gain an intuitive insight into whether the average argument vectors behave as expected, we show in Table 1 the top five most similar words to the average argument vectors for several roles and predicates. When evaluated with the data sets described in Section 2, this initial method leads to 61.32% semantic F-score for Finnish and 65.05% for English.

3.3 Incorporating syntax

As we will demonstrate shortly, incorporating information about dependency relations can lead to a substantial performance gain. To incorporate the dependency relation information into the role classification method introduced above, we apply the technique of circular shifting of vectors. This technique was previously used in the context of *Random Indexing (RI)* to derive new vectors from existing ones in a deterministic fashion (Basile and Caputo, 2012). In RI, the shift operation is however not used on the final vectors, but rather already during the induction of the vector representation.

Given a vector representation of an argument y, we can encode the dependency relation of y and its predicate by circularly shifting the vector of y by an offset assigned separately to each possible dependency relation. The assignment is arbitrary, but such that no two relations are assigned the same offset. We will denote this operation as $y{\gg}d$, meaning the vector y circularly shifted to the right by the offset assigned to the dependency relation d. For instance, circularly shifting a vector $a = (1, 2, 3, 4, 5)$ to the right by an offset of 2 results in $a{\gg}2 = (4, 5, 1, 2, 3)$.

We can incorporate the dependency relations when calculating the average vectors representing arguments as follows:

$$A(x, r) = \frac{\sum_{(r,d,x,y)} \hat{y}{\gg}d}{count}\ , \qquad (3)$$

where (r, d, x, y) iterates over all predicate-argument pairs (x, y) where y has the dependency relation d and role r. The role of an argument in the test phase is established as before, by taking the role which maximizes the similarity to the average vector:

$$\arg\max_r sim(A(x, r), y{\gg}d) \qquad (4)$$

In the cases, where arguments are not direct dependents of the predicate, we use zero as the shift offset.

To motivate this approach and illustrate its implications, consider the two sentences (1) *The cat chases the dog.* (2) *The dog chases the cat.* In the first sentence the *dog* is an object which corresponds to the theme role *A1*, whereas in the second sentence it is a subject with the agent role *A0*. The role labeling decision is, however, in both cases based on the similarity value $sim(A(chases, r), dog)$, predicting *A1*, which is incorrect in the latter case. When we incorporate the syntactic information by shifting the vector according to its syntactic relation to the predicate, we obtain two diverging similarity values because $dog \gg nsubj$ and $dog \gg dobj$ are essentially two different vectors. This leads to the correct prediction in both cases.

Relative to the base method, incorporating the syntax improves the semantic F-score from 61.32% to 66.23% for Finnish and from 65.05% to 66.55% for English. For Finnish, the gain is rather substantial, while for English we see only a moderate but nevertheless positive effect. This demonstrates that, indeed, the circular shifting operation successfully encodes syntactic information both into the average vectors A and the candidate argument vectors y.

3.4 Core arguments vs. modifiers

In comparison to modifier roles, the assignment of core (numbered) argument roles is considerably more influenced by the predicate sense and therefore must be learned separately, which we also confirmed in initial experiments. The modifier roles, on the other hand, are global in the sense that they are not tied to any particular predicate. This brings out an interesting question of whether the modifier roles should be learned independently of the predicate or not. We find that the best strategy is to learn predicate-specific modifier vectors in English and global modifier vectors in Finnish.

Another problem, particularly common in the Finnish PropBank stems from the distinction between core roles and modifier roles. For instance, for the predicate *to move* the argument meaning the destination of the moving action has the core role *A2*, while for a number of other predicates which may optionally take a destination argument, the directional modifier role *AM-DIR* would be used. This leads to a situation where core arguments receive a high score for a modifier role, and modifier roles are over-predicted at the expense of core argument roles. To account for this, we introduce the following simple heuristics. If the predicate lacks a core role r after prediction, iterate through predicted modifier roles $p_1 \ldots p_n$ and change the prediction from p_i to r if r has the maximum similarity among the core roles and the difference $sim(p_i, y) - sim(r, y)$ is smaller than a threshold value optimized on a held-out develop-

ment set distinct from the test set.

We observe a 2.05pp gain in Finnish when using this method, whereas in English this feature is less significant with an improvement of only 0.3pp.

3.5 Fall-back for unknown words

The above-mentioned techniques based purely on vector representations with no additional features fail if the vector space model lacks the argument token which prevents the calculation of the necessary similarities. To address this problem, we build separately for each POS a "generic" representation by averaging the vectors of all training data tokens that have the POS and occurred only once. These vectors, representing a typical rare word of a given POS, are then used in place of words missing from the vector space model.

Another solution taking advantage from the vector space representation is used in cases where a predicate is not seen in the training data and therefore we have no information about its argument structure. We query for predicates closest to the unseen predicate and take the average argument vectors from the most similar predicate that was seen during the training.

Together, these two techniques result in a modest gain of approximately 1pp for both languages.

3.6 Sense classification

One final step required in SRL is the disambiguation of the sense of the predicate. Here we apply an approach very similar to that used for role classification, whereby for every sense of every predicate, we calculate an average vector representing that sense. This is done as follows: For every predicate sense, we average the vector representations of all dependents and governors[7] of all occurrences of that sense in the training data, circularly shifted to encode their syntactic relation to the predicate. To assign a sense to a predicate during testing, we average the shifted vectors corresponding to its dependents and governors in the sentence, and choose the sense whose average vector is the nearest. Using this approach, we obtain a 84.18% accuracy for Finnish and 92.68% for English, compared to 79.89% and 92.88% without the syntax information. This corresponds to a substantial gain for Finnish but, surprisingly, a small drop for English. For the rare predicates that are not seen in the training data, we have no information about their sense inventory and therefore we simply predict the sense ".01" which is the correct choice in 79.56% of the cases in Finnish and 86.64% in English.

[7]Recall we use the extended SD scheme where a word can have several governors in various situations.

4 Role Labeling with Linear Transformations

As we discussed earlier, it was recently shown that the *word2vec* spaces preserve a number of linguistic regularities, and an accurate mapping between two *word2vec*-induced vector spaces can be achieved using a simple linear transformation. Mikolov et al. (2013a) have demonstrated that a linear transformation trained on source-target language word pairs obtained from Google Translate can surprisingly accurately map word vectors from one language to another, with obvious applications to machine translation. It is also worth noting that this is not universally true of all vector space representation methods, as Mikolov et al. have shown for example for Latent Semantic Analysis, which exhibits this property to a considerably lesser extent. In addition to testing the applicability of the *word2vec* method in general, we are specifically interested whether these additional properties can be exploited in the context of SRL. In particular, we will test whether a similar linear vector space transformation can be used to map the vectors of the predicates onto those of their arguments.

More formally, for each role r, we will learn a transformation matrix W_r such that for a vector representation x of some predicate, $W_r x$ will be close to the vector representation of its arguments with the role r. For instance, if x is the representation of the predicate *(to) eat*, we aim for $W_{A1}x$ to be a vector similar to the vectors representing edible items (role *A1*). The transformation can be trained using the tuples (r, x, y) of predicate x and its argument y with the role r gathered from the training data, minimizing the error

$$\sum_{(x,y)} \|W_r x - y\|^2 \tag{5}$$

over all training pairs (separately for each r). We minimize the error using the standard stochastic gradient descent method, whereby the transformation matrix is updated separately for each pair (x, y) using the equation

$$W_r \leftarrow W_r - \epsilon(W_r x - y)x^T \tag{6}$$

where ϵ is the learning rate whose suitable value is selected on the development set. The whole procedure is repeated until convergence is reached, randomly shuffling the training data after each round of training.

Using the transformation, we can establish the most likely role for the argument y of a predicate x as

$$\arg\max_r sim(W_r x, y) \quad (7)$$

where sim is the cosine similarity function, i.e. in the exact same manner as for the average argument method described in the previous section, with the difference that the vector for the average argument is not calculated directly from the training data, but rather obtained through the linear transformation of the predicate vector.

As an alternative approach, we can also learn the reverse transformation RW_r such that $RW_r y$ is close to x, i.e. the transformation of the argument y onto the predicate x. Note that here RW_r is not the same as W_r^T; we train this reverse transformation separately using the same gradient descent method. We then modify the method for finding the most likely role for an argument by taking the average of the forward and reverse transformation similarities:

$$\arg\max_r \frac{sim(W_r x, y) + sim(x, RW_r y)}{2} \quad (8)$$

Note that we make no assumptions about the vector spaces where x and y are drawn from; they may be different spaces and they do not need to be matched in their dimensionality either, as there is no requirement that W and RW be square matrices. In practice, we find that the best strategy for both Finnish and English is to represent both the predicates and arguments using the space induced from word forms, however, we have also tested on Finnish representing the predicates using the space induced from lemmas and the arguments using a space induced from word forms, with only minimally worse results. This shows that the transformation does not degrade substantially even when mapping between two different spaces.

With this strategy, we reach an F-score of 62.71% in Finnish and 63.01% in English. These results are on par with the scores obtained with the average argument method, showing that a linear transformation is effective also in this kind of problems.

To incorporate syntax information, we train transformation matrices $W_{r,d}$ and $RW_{r,d}$ for each dependency relation d rather than relying on the circular shift operation which cannot be captured by linear transformations.[8] As some combinations of r and d may occur only in the test data, we use the matrices W_r and RW_r as a fall-back strategy. In testing, we found that even if the (r, d) combination is known from the training data, a small improvement can be obtained by taking the average of the similarities with and without syntactic information as the final similarity. Incorporating these techniques into the basic linear transformation improves the semantic F-score from 62.71% to 65.88% for Finnish and from 63.01% to 67.04% for English. The improvement for both languages is substantial.

5 Supervised classification approach

In the previous sections, we have studied an approach to SRL based purely on the vector space representations with no additional features. We have addressed the choice of the argument role by simply assigning the role with the maximum similarity to the argument. To test the gain that could be obtained by employing a more advanced technique for aggregating the scores and incorporating additional features, we train a linear multi-class support vector machine to assign the role to every detected argument. As features, we use the similarity values for each possible role using the best performing method for each language,[9] the sense of the predicate, and — separately for the predicate and the argument — the token itself, its POS, morphological tags, every dependency relation to its governors, and every dependency relation to its dependents. The similarities are encoded as feature weights, while all other features are binary. We use the multi-class SVM implementation from the SVM-multiclass package (Joachims, 1999), setting the regularization parameter on the development set using a grid search.

For both languages, we observe a notable gain in performance, leading to the best scores so far. In Finnish the improvement in F-score is from 66.23% to 73.83% and in English from 67.04% to 70.38%. However, as we will further discuss in Section 6, in Finnish the contribution of the similarity features is modest.

[8] Note that this does not affect the overall computational cost, as the total number of training examples remains unchanged and the transformation matrices are small in size.

[9] Average vectors for Finnish and transformation for English (Table 2).

	Finnish	English
Average vectors		
full method	66.23	66.55
–modifier vs. core role	64.18	66.25
–syntax	61.32	65.05
Linear transformation		
full method	65.88	67.04
–syntax	62.71	63.01
Supervised classification		
full method	73.83	70.38
only similarity features	64.21	65.89
–similarity features	73.54	67.51
–lexical features	65.51	58.42

Table 2: Overview of main results and a feature ablation study. *Modifier vs. core role* refers to the algorithm presented in Section 3.4. In the supervised classification part, *-lexical features* refers to the removal of features based on word forms, predicate sense and role similarities.

6 Results and discussion

All results discussed throughout Sections 3 to 5 are summarized in Table 2, which also serves as a coarse feature ablation study. Overall, we see that the average vector and linear transformation methods perform roughly on par, with the average vector method being slightly better for Finnish and slightly worse for English. Both vector space-based methods gain notably from syntax information, confirming that the manner in which this information is incorporated is indeed meaningful.

Adding the SVM classifier on top of these two methods results in a substantial further raise in performance, demonstrating that to be competitive on SRL, it is necessary to explicitly model also additional information besides the semantic similarity between the predicate and the argument. This is particularly pronounced for Finnish where the present SVM method does not gain substantially from the similarity-based features, while English clearly benefits. To shed some light on this difference, we show in Table 3 the oracle accuracy of role labeling for top-1 through top-10 roles as ordered by their similarity scores. The performance on English is clearly superior to that on Finnish. An important factor may be the fact that — in terms of token count — the CoNLL'09 English training size is nearly six times that of the Finnish PropBank and the English vector space model was induced on a nearly four times larger text corpus.

Finnish		English	
n	Recall	n	Recall
1	58.23	1	67.39
2	68.29	2	82.82
3	74.30	3	88.49
4	78.71	4	91.58
5	82.21	5	93.20
6	84.74	6	94.29
7	87.04	7	94.91
8	88.98	8	95.31
9	90.76	9	95.65
10	92.05	10	95.86

Table 3: A study of how many times (%) the correct role is among the top n most similar roles when the arguments are known in advance. Left: Similarities taken from the average vector method on Finnish. Right: Similarities from the linear transformation method on English.

	Finnish	English
Average vectors	66.23 / 89.89	66.55 / 79.57
Linear transf.	65.88 / 89.92	67.04 / 80.85
Supervised	73.83 / 89.29	70.38 / 78.71

Table 4: Overall results separately for all main methods (labeled/unlabeled semantic F-score).

Returning to our original question of whether the SD scheme can be used as-is for argument identification, we show in Table 4 the unlabeled F-scores for the main methods. These scores reflect the performance of the argument identification step in isolation. While Finnish approaches 90% which is comparable to the best systems in the CoNLL'09 task, English lags behind by over 10pp. To test to what extent the results would be affected if a more accurate argument identification system was applied, we used the output of the winning (for English) CoNLL'09 Shared Task system (Zhao et al., 2009a) as the argument identification component, while predicting the roles with the methods introduced so far. The results are summarized in Table 5, where we see a substantial gain for all the methods presented in this paper, achieving an F-score of 82.33%, only 3.82pp lower than best CoNLL'09 system. These results give a mixed signal as to whether the extended SD scheme is usable nearly as-is for argument identification (Finnish) or not (English). Despite our efforts, we were unable to pinpoint the cause for this difference, beyond the fact that the Finnish Prop-

	Semantic F-score
CoNLL'09 best	86.15 / 91.97
Average vectors	73.12 / 91.97
Linear transformation	74.41 / 91.97
Supervised classif.	82.33 / 91.97

Table 5: Performance of suggested methods with argument identification from the top-performing CoNLL'09 system (labeled/unlabeled F-score).

Bank was originally developed specifically on top of the SD scheme, while the English PropBank and NomBank corpora were not.

7 Related work

While different methods have been studied to build task specific vector space representations, post-hoc methods to manipulate the vector spaces without retraining are rare. Current SRL systems utilize supervised machine learning approaches, and typically a large set of features. For instance, the winning system in the CoNLL'09 shared task (SRL-only) introduces a heavy feature engineering system, which has about 1000 potential feature templates from which the system discovers the best set to be used (Zhao et al., 2009b). Word similarities are usually introduced to SRL as a part of unsupervised or semi-supervised methods. For example, Titov and Klementiev (2012) present an unsupervised clustering method applying word representation techniques, and Deschacht and Moens (2009) used vector similarities to automatically expand the small training set to build semi-supervised SRL system. Additionally, Turian et al. (2010) have shown that word representations can be included among the features to improve the performance of named entity recognition and chunking systems.

8 Conclusions

We set out to test two post-hoc vector space manipulation techniques in the context of semantic role labeling. We found that the circular shift operation can indeed be applied also to other vector representations as a way to encode syntactic information. Importantly, the circular shift is applied to a pre-existing vector space representation, rather than during its induction, and is therefore task-independent. Further, we find that such shifted vectors can be meaningfully averaged to represent predicate senses and arguments.

We also extended the study of the linear transformation between two vector spaces and show that the same technique can be used also within a single space, mapping the vectors of predicates onto the vectors of their arguments. This mapping produces results that are performance-wise on par with the average vectors method, demonstrating a good generalization ability of the linear mapping and the underlying *word2vec* vector space representation. Here it is worth noting that — if we gloss over some obvious issues of ambiguity — the mapping between two languages demonstrated by Mikolov et al. is conceptually a one-to-one mapping, at least in contrast to the one-to-many nature of the mapping between predicates and their arguments. These results hint at the possibility that a number of problems which can be reduced to the "predict a word given a word" pattern may be addressable with this simple technique.

With respect to the application to SRL, we have shown that it is possible to carry out SRL based purely on the vector space manipulation methods introduced in this paper, outperforming several entries in the CoNLL-09 Shared Task. However, it is perhaps not too surprising that much more is needed to build a competitive SRL system. Adding an SVM classifier with few relatively simple features derived from the syntactic analyses in addition to features based on vector similarities, and especially adding a well-performing argument identification method, can result in a system close to approaching state-of-the-art performance, which is encouraging.

As future work, it will be interesting to study to which extent SRL, and similar applications would benefit from addressing the one-to-many nature of the underlying problem. While for some predicates the arguments likely form a cluster that can be represented as a single average vector, for other predicates, such as *to see*, it is not the case. Finding methods which allow us to model this property of the problem will constitute an interesting direction with broader applications beyond SRL.

Acknowledgements

This work has been supported by the Emil Aaltonen Foundation and Kone Foundation. Computational resources were provided by CSC – IT Center for Science. We would also like to thank Sampo Pyysalo and Hans Moen for comments and general discussion.

References

Pierpaolo Basile and Annalina Caputo. 2012. Encoding syntactic dependencies using Random Indexing and Wikipedia as a corpus. In *Proceedings of the 3rd Italian Information Retrieval (IIR) Workshop*, volume 835, pages 144–154.

Bernd Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 89–97. Association for Computational Linguistics.

Koen Deschacht and Marie-Francine Moens. 2009. Semi-supervised semantic role labeling using the latent words language model. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 21–29. Association for Computational Linguistics.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, et al. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–18. Association for Computational Linguistics.

Péter Halácsy, András Kornai, and Csaba Oravecz. 2007. HunPos: an open source trigram tagger. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 209–212. Association for Computational Linguistics.

Katri Haverinen, Veronika Laippala, Samuel Kohonen, Anna Missilä, Jenna Nyblom, Stina Ojala, Timo Viljanen, Tapio Salakoski, and Filip Ginter. 2013a. Towards a dependency-based PropBank of general Finnish. In *Proceedings of the 19th Nordic Conference on Computational Linguistics (NoDaLiDa'13)*, pages 41–57.

Katri Haverinen, Jenna Nyblom, Timo Viljanen, Veronika Laippala, Samuel Kohonen, Anna Missilä, Stina Ojala, Tapio Salakoski, and Filip Ginter. 2013b. Building the essential resources for Finnish: the Turku Dependency Treebank. *Language Resources and Evaluation*, pages 1–39.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. MIT Press.

Krister Lindén, Miikka Silfverberg, and Tommi Pirinen. 2009. HFST tools for morphology — an efficient open-source package for construction of morphological analyzers. In *State of the Art in Computational Morphology*, volume 41 of *Communications in Computer and Information Science*, pages 28–47.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8. Coling 2008 Organizing Committee.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank project: An interim report. In *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, pages 24–31.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. *CoRR (arxiv.org)*, abs/1309.4168.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751. Association for Computational Linguistics, June.

Jenna Nyblom, Samuel Kohonen, Katri Haverinen, Tapio Salakoski, and Filip Ginter. 2013. Predicting conjunct propagation and other extended Stanford Dependencies. In *Proceedings of the International Conference on Dependency Linguistics (Depling 2013)*, pages 252–261.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Tommi Pirinen. 2008. Suomen kielen äärellistilainen automaattinen morfologinen jäsennin avoimen lähdekoodin resurssein. Master's thesis, University of Helsinki.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. BRAT: a web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107. Association for Computational Linguistics.

Ivan Titov and Alexandre Klementiev. 2012. A Bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 12–22. Association for Computational Linguistics.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.

Hai Zhao, Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009a. Multilingual dependency learning: Exploiting rich features for tagging syntactic and semantic dependencies. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 61–66. Association for Computational Linguistics.

Hai Zhao, Wenliang Chen, Chunyu Kit, and Guodong Zhou. 2009b. Multilingual dependency learning: a huge feature engineering method to semantic dependency parsing. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 55–60. Association for Computational Linguistics.