A Neural Network Approach to Selectional Preference Acquisition

Tim Van de Cruys
IRIT & CNRS
Toulouse, France
tim.vandecruys@irit.fr

Abstract

This paper investigates the use of neural networks for the acquisition of selectional preferences. Inspired by recent advances of neural network models for NLP applications, we propose a neural network model that learns to discriminate between felicitous and infelicitous arguments for a particular predicate. The model is entirely unsupervised – preferences are learned from unannotated corpus data. We propose two neural network architectures: one that handles standard two-way selectional preferences and one that is able to deal with multi-way selectional preferences. The model's performance is evaluated on a pseudo-disambiguation task, on which it is shown to achieve state of the art performance.

1 Introduction

Predicates often have a semantically motivated preference for particular arguments. Compare for example the sentences in (1) and (2).

(1) The vocalist sings a ballad.

(2) The exception sings a tomato.

Most language users would have no problems accepting the first sentence as well-formed: a vocalist can be expected to sing, and a ballad is something that can be sung. The same language users, however, would likely consider the second sentence to be ill-formed: an exception is not supposed to sing, nor is a tomato something that is typically sung. Within the field of natural language processing, this inclination of predicates to select for particular arguments is known as *selectional preference*.

The automatic acquisition of selectional preferences has been a popular research subject within the field of natural language processing. An automatically acquired selectional preference resource is a versatile tool for numerous NLP applications, such as semantic role labeling (Gildea and Jurafsky, 2002), word sense disambiguation (McCarthy and Carroll, 2003), and metaphor processing (Shutova et al., 2013).

Models for selectional preference need to adequately deal with the consequences of Zipf's law: language is inherently sparse, and the majority of language utterances occur very infrequently. As a consequence, models that are based on corpus data need to properly generalize beyond the mere co-occurrence frequencies of sparse corpus data, taking into account the semantic similarity of both predicates and arguments. Researchers have come up with various approaches to this generalization step. Earlier approaches to selectional preference acquisition mostly rely on hand-crafted resources such as WordNet (Resnik, 1996; Li and Abe, 1998; Clark and Weir, 2001), while later approaches tend to take advantage of unsupervised learning machinery, such as latent variable models (Rooth et al., 1999; Ó Séaghdha, 2010) and distributional similarity metrics (Erk, 2007; Padó et al., 2007).

This paper investigates the use of neural networks for the acquisition of selectional preferences. Inspired by recent advances of neural network models for NLP applications (Collobert and Weston, 2008; Mikolov et al., 2013), we propose a neural network model that learns to discriminate between felicitous and infelicitous arguments for a particular predicate. The model is entirely unsupervised – preferences are learned from unannotated corpus data. Positive training instances are constructed from attested corpus data, while negative instances are constructed from randomly corrupted instances. We propose two neural network architectures: one that handles standard two-way selectional preferences and one that is able to deal with multi-way selectional preferences, where the interaction be-

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 26–35,
October 25-29, 2014, Doha, Qatar. ©2014 Association for Computational Linguistics

tween multiple verb arguments is taken into account. The model's performance is evaluated on a pseudo-disambiguation task, on which it is shown to achieve state of the art performance.

The contributions of this paper are twofold. First of all, we apply and evaluate a neural network approach to the problem of standard (two-way) selectional preference acquisition. Selectional preference acquisition using neural networks has not yet been explored in the literature. Secondly, we propose a novel network architecture and training objective for the acquisition of multi-way selectional preferences, where the interaction between a verb and its various arguments is captured at the same time.

The remainder of this paper is as follows. Section 2 first discusses related work with respect to selectional preference acquisition and neural network modeling. Section 3 describes our neural network architecture and its training procedure. Section 4 evaluates the model's performance, comparing it to other existing models for selectional preference acquisition. Finally, section 5 concludes and indicates a number of avenues for future work.

2 Related Work

2.1 Selectional preferences

One of the first approaches to the automatic induction of selectional preferences from corpora was the one by Resnik (1996). Resnik (1996) relies on WordNet synsets in order to generate generalized noun clusters. The *selectional preference strength* of a specific verb v in a particular relation is calculated by computing the Kullback-Leibler divergence between the cluster distribution of the verb and the prior cluster distribution.

$$S_{R(v)} = \sum_{c} \mathrm{p}(c|v) \log \frac{\mathrm{p}(c|v)}{\mathrm{p}(c)} \tag{1}$$

where c stands for a noun cluster, and R stands for a given predicate-argument relation. The *selectional association* of a particular noun cluster is then the contribution of that cluster to the verb's preference strength.

$$A_{R(v,c)} = \frac{\mathrm{p}(c|v) \log \frac{\mathrm{p}(c|v)}{\mathrm{p}(c)}}{S_{R(v)}} \tag{2}$$

The model's generalization relies entirely on WordNet, and there is no generalization among the verbs.

Other researchers have equally relied on WordNet in order to generalize over arguments. Li and Abe (1998) use the principle of Minimum Description Length in order to find a suitable generalization level within the lexical WordNet hierarchy. A same intuition is used by Clark and Weir (2001), but they use hypothesis testing instead to find the appropriate level of generalization. A recent approach that makes use of WordNet (in combination with Bayesian modeling) is the one by Ó Séaghdha and Korhonen (2012).

Most researchers, however, acknowledge the shortcomings of hand-crafted resources, and focus on the acquisition of selectional preferences from corpus data. Rooth et al. (1999) propose an Expectation-Maximization (EM) clustering algorithm for selectional preference acquisition based on a probabilistic latent variable model. The idea is that both predicate v and argument o are generated from a latent variable c, where the latent variables represent clusters of tight verb-argument interactions.

$$\mathrm{p}(v,o) = \sum_{c \in C} \mathrm{p}(c,v,o) = \sum_{c \in C} \mathrm{p}(c)\mathrm{p}(v|c)\mathrm{p}(o|c) \tag{3}$$

The use of latent variables allows the model to generalize to predicate-argument tuples that have not been seen during training. The latent variable distribution – and the probabilities of predicates and argument given the latent variables – are automatically induced from data using EM. We will compare against their model for evaluation purposes.

Erk (2007) and Erk et al. (2010) describe a method that uses corpus-driven distributional similarity metrics for the induction of selectional preferences. The key idea is that a predicate-argument tuple (v,o) is felicitous if the predicate v appears in the training corpus with arguments o' that are similar to o, i.e.

$$S(v,o) = \sum_{o' \in O_v} \frac{wt(v,o')}{Z(v)} \cdot sim(o,o') \tag{4}$$

where O_v represents the set of arguments that have been attested with predicate v, $wt(\cdot)$ represents an appropriate weighting function (such as the frequency of the (v,o') tuple), and Z is a normalization factor. We equally compare to their model for evaluation purposes.

Bergsma et al. (2008) present a discriminative approach to selectional preference acquisition. Positive examples are taken from observed predicate-argument pairs, while negative examples are constructed from unobserved combinations. An SVM classifier is used to distinguish the positive from the negative instances. The training procedure used in their model is based on an intuition that is similar to ours, although it is implemented using different techniques.

A number of researchers presented models that are based on the framework of topic modeling. Ó Séaghdha (2010) describes three models for selectional preference induction based on Latent Dirichlet Allocation, which model the selectional preference of a predicate and a single argument. Ritter et al. (2010) equally present a selectional preference model based on topic modeling, but they tackle multi-way selectional preferences (of transitive predicates, which take two arguments) instead.

Finally, in previous work (Van de Cruys, 2009) we presented a model for multi-way selectional preference induction based on tensor factorization. Three-way co-occurrences of subjects, verbs, and objects are represented as a three-way tensor (the generalization of a matrix), and a latent factorization model is applied in order to generalize to unseen instances. We will compare our neural network based-approach for multi-way selectional preference acquisition to this tensor-based factorization model.

2.2 Neural networks

In the last few years, neural networks have become increasingly popular in NLP applications. In particular, neural language models (Bengio et al., 2003; Mnih and Hinton, 2007; Collobert and Weston, 2008) have demonstrated impressive performance at the task of language modeling. By incorporating distributed representations for words that model their similarity, neural language models are able to overcome the problem of data sparseness that standard n-gram models are confronted with. Also related to our work is the approach by Tsubaki et al. (2013), who successfully use a neural network to model co-compositionality.

Our model for selectional preference acquisition uses a network architecture that is similar to the abovementioned models. Its training objective is also similar to the ranking-loss training objective proposed by Collobert and Weston (2008), but we present a novel, modified version in order to deal with multi-way selectional preferences.

3 Methodology

3.1 Neural network architecture

Our model computes the score for a predicate i and an argument j as follows. First, the selectional preference tuple (i,j) is represented as the concatenation of the vectors $\mathbf{v}_i$ and $\mathbf{o}_j$, i.e.

$$\mathbf{x} = [\mathbf{v}_i, \mathbf{o}_j] \tag{5}$$

Vectors $\mathbf{v}_i$ and $\mathbf{o}_j$ are extracted from two embedding matrices, $\mathbf{V} \in \mathbb{R}^{N \times I}$ (the predicate matrix, where I represents the number of elements in the predicate vocabulary) and $\mathbf{O} \in \mathbb{R}^{N \times J}$ (the argument matrix, where J represents the number of elements in the argument vocabulary). N is a parameter setting of the model, representing the vector size of the embeddings. Matrices $\mathbf{V}$ and $\mathbf{O}$ will be automatically learned during training.

Vector $\mathbf{x}$ then serves as input vector to our neural network. We use a feed-forward neural network architecture with one hidden layer:

$$\begin{aligned} \mathbf{a}_1 &= f(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) && (6) \\ y &= \mathbf{W}_2\mathbf{a}_1 && (7) \end{aligned}$$

where $\mathbf{x} \in \mathbb{R}^{2N}$ is our input vector, $\mathbf{a}_1 \in \mathbb{R}^{H}$ represents the activation of the hidden layer with H hidden nodes, $\mathbf{W}_1 \in \mathbb{R}^{H \times 2N}$ and $\mathbf{W}_2 \in \mathbb{R}^{1 \times H}$ respectively represent the first and second layer weights, $\mathbf{b}_1$ represents the first layer's bias, $f(\cdot)$ represents the element-wise activation function tanh, and y is our final selectional preference score. The left-hand picture of figure 1 gives a graphical representation of our standard neural network architecture.

3.2 Training the network

A proper estimation of a neural network's parameters requires a large amount of training data. To be able to use non-annotated corpus data for training, we use the method proposed by Collobert and Weston (2008). The authors present a method for training a neural network language model from unlabeled data by corrupting actual attested n-grams with a random word. They then define a ranking-type cost function, which allows the network to learn to discriminate between good and bad word sequences. We adopt the same method for our selectional preference model as follows.

Let (i,j) be our proper, attested predicate-argument tuple. The goal of our model is to discriminate the correct tuple (i,j) from other, non-attested tuples (i,j'), in which the correct predicate

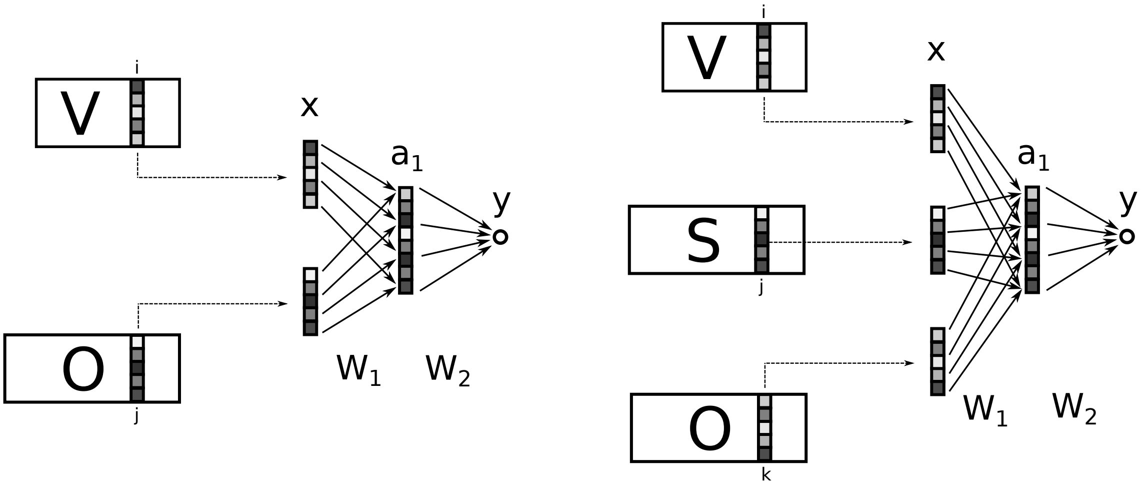

Figure 1: Neural network architectures for selectional preference acquisition. The left-hand picture shows the architecture for two-way selectional preferences, the right-hand picture shows the architecture for three-way selectional preferences. In both cases, vector **x** is constructed from the appropriate predicate and argument vectors from the embedding matrices, and fed forward through the network to yield a preference score *y*.

j has been replaced with a random predicate j'. We require the score for the correct tuple to be larger than the score for the corrupt tuple by a margin of one. For one tuple (i,j), this corresponds to minimizing the objective function in (8)

$$\sum_{j' \in J} \max(0, 1 - g[(i,j)] + g[(i,j')]) \qquad (8)$$

where J represents the predicate vocabulary, and $g[\cdot]$ represents our neural network scoring function presented in the previous section.

In line with Collobert and Weston (2008), the gradient of the objective function is sampled by randomly picking one corrupt argument j' from the argument vocabulary for each attested predicate-argument tuple (i,j). The derivative of the cost with respect to the model's parameters (weight matrices $\mathbf{W}_1$ and $\mathbf{W}_2$, bias vector $\mathbf{b}_1$, and embedding matrices **V** and **O**) is computed, and the appropriate parameters are updated through backpropagation.

3.3 Multi-way selectional preferences

The model presented in the previous section is only able to deal with two-way selectional preferences. In this section, we present an extension of the model that is able to handle multi-way selectional preferences.[1]

In order to model the selectional preference of a transitive verb for its subject and direct object, we start out in a similar fashion to the two-way case. Instead of having only one embedding matrix, we now have two embedding matrices $\mathbf{S} \in \mathbb{R}^{N \times J}$ and $\mathbf{O} \in \mathbb{R}^{N \times K}$, representing the two different argument slots of a transitive predicate. Our input vector can now be represented as

$$\mathbf{x} = (\mathbf{v}_i, \mathbf{s}_j, \mathbf{o}_k) \qquad (9)$$

Note that $\mathbf{x} \in \mathbb{R}^{3N}$ and $\mathbf{W}_1 \in \mathbb{R}^{H \times 3N}$. The rest of our neural network architecture stays exactly the same. The right-hand picture of figure 1 presents a graphical representation.

For the multi-way case, we present an adapted version of the training objective. Given an attested subject-verb-object tuple (i,j,k), the goal of our network is now to discriminate this correct tuple from other, corrupted tuples (i,j,k'), (i,j',k) and (i,j',k'), where the correct arguments have been replaced by random subjects j' and random objects k'. Note that we do not only want the network to learn the infelicity of tuples in which both the subject and object slot are corrupted; we also want our network to learn the infelicity of tuples in which either the subject or object slot is corrupt, while the other slot contains the correct, attested argument. This leads us to the objective function represented in (10).

[1] We exemplify the model using three-way selectional preferences for transitive predicates, but the model can be straightforwardly generalized to other multi-way selectional preferences.

$$\begin{aligned}&\sum_{k'\in K}\max(0,1-g[(i,j,k)]+g[(i,j,k')])\\&+\sum_{j'\in J}\max(0,1-g[(i,j,k)]+g[(i,j',k)])\\&+\sum_{\substack{j'\in J\\k'\in K}}\max(0,1-g[(i,j,k)]+g[(i,j',k')])\end{aligned}\tag{10}$$

As in the two-way case, the gradient of the objective function is sampled by randomly picking one corrupted subject j' and one corrupted object k' for each tuple (i,j,k). All of the model's parameters are again updated through backpropagation.

4 Evaluation

4.1 Implementational details

We evaluate our neural network approach to selectional preference acquisition using verb-object tuples for the two-way model, and subject-verb-object tuples for the multi-way model.

Our model has been applied to English, using the UKWaC corpus (Baroni et al., 2009), which covers about 2 billion words of web text. The corpus has been part of speech tagged and lemmatized with Stanford Part-Of-Speech Tagger (Toutanova et al., 2003), and parsed with MaltParser (Nivre et al., 2006), so that dependency tuples could be extracted.

For the two-way model, we select all verbs and objects that appear within a predicate-argument relation with a frequency of at least 50. This gives us a total of about 7K verbs and 30K objects. For the multi-way model, we select the 2K most frequent verbs, together with the 10K most frequent subjects and the 10K most frequent objects (that appear within a transitive frame).

All words are converted to lowercase. We use the lemmatized forms, and only keep those forms that contain alphabetic characters. Furthermore, we require each tuple to appear at least three times in the corpus.

We set N, the size of our embedding matrices, to 50, and H, the number of units in the hidden layer, to 100. Following Huang et al. (2012), we use mini-batch L-BFGS (Liu and Nocedal, 1989) with 1000 pairs of good and corrupt tuples per batch for training, and train for 10 epochs.

4.2 Evaluation Setup

4.2.1 Task

Our models are quantitatively evaluated using a pseudo-disambiguation task (Rooth et al., 1999), which bears some resemblance to our training procedure. The task provides an adequate test of the generalization capabilities of our models. For the two-way case, the task is to judge which object (o or o') is more likely for a particular verb v, where (v,o) is a tuple attested in the corpus, and o' is a direct object randomly drawn from the object vocabulary. The tuple is considered correct if the model prefers the attested tuple (v,o) over (v,o'). For the three-way case, the task is to judge which subject (s or s') and direct object (o or o') are more likely for a particular verb v, where (v,s,o) is the attested tuple, and s' and o' are a random subject and object drawn from their respective vocabularies. The tuple is considered correct if the model prefers the attested tuple (v,s,o) over the alternatives (v,s,o'), (v,s',o), and (v,s',o'). Tables 1 and 2 respectively show a number of examples from the two-way and three-way pseudo-disambiguation task.

v	o	o'
perform	*play*	*geometry*
buy	*wine*	*renaissance*
read	*introduction*	*peanut*

Table 1: Pseudo-disambiguation examples for two-way verb-object tuples

v	s	o	s'	o'
win	*team*	*game*	*diversity*	*egg*
publish	*government*	*document*	*grid*	*priest*
develop	*company*	*software*	*breakfast*	*landlord*

Table 2: Pseudo-disambiguation examples for three-way subject-verb-object tuples

The models are evaluated using 10-fold cross validation. All tuples from our corpus are randomly divided into 10 equal parts. Next, for each fold, 9 parts are used for training, and the remaining part is used for testing. In order to properly test the generalization capability of our models, we make sure that all instances of a particular tuple appear in one part only. This way, we make sure that tuples used for testing are never seen during training.

For the two-way model, our corpus consists of about 70M tuple instances (1.9M types), so in each fold, about 63M tuple instances are used for training and about 7M (190K types) are used for testing. For the three-way model, our corpus consists of about 5,5M tuple instances (750K types), so in each fold, about 5M tuples are used for training and about 500K (75K types) are used for testing. Note that our training procedure is instance-based, while our evaluation is type-based: during training, the neural network sees a tuple as many times as it appears in the training set, while for testing each individual tuple is only evaluated once.

4.2.2 Comparison models

We compare our neural network model to a number of other models for selectional preference acquisition.

For the two-way case, we compare our model to the EM-based clustering technique presented by Rooth et al. (1999),[2] and to Erk et al.'s (2010) similarity-based model. For Rooth et al.'s model, we set the number of latent factors to 50. Using a larger number of latent factors does not increase performance. For Erk et al.'s model, we create a dependency-based similarity model from the UKWaC corpus using our 30K direct objects as instances and 100K dependency relations as features. The resulting matrix is weighted using pointwise mutual information (Church and Hanks, 1990). Similarity values are computed using cosine. Furthermore, we use a sampling procedure in the testing phase: we sample 5000 predicate-argument pairs for each fold, as testing Erk et al.'s model on the complete test sets proved prohibitively expensive.

For the three-way case, we compare our model to the tensor factorization model we developed in previous work (Van de Cruys, 2009). We set the number of latent factors to 300.[3]

4.3 Results

4.3.1 Two-way model

Table 3 compares the results of our neural network architecture for two-way selectional preferences to the results of Rooth et al.'s (1999) model and Erk et al.'s (2010) model.

model	accuracy ($\mu \pm \sigma$)
Rooth et al. (1999)	.720 ± .002
Erk et al. (2010)	**.887** ± .004
2-way neural network	.880 ± .001

Table 3: Comparison of model results for two-way selectional preference acquisition – mean accuracy and standard deviations of 10-fold cross-validation results

The results indicate that our neural network approach outperforms Rooth et al.'s (1999) method by a large margin (16%). Clearly, the neural network architecture is able to model selectional preferences more profoundly than Rooth et al.'s latent variable approach. The difference between the models is highly statistically significant (paired t-test, $p < .01$), as the standard deviations already indicate.

Erk et al.'s model reaches a slightly better score than our model, and this result is also statistically significant (paired t-test, $p < .01$). However, Erk et al.'s model does not provide full coverage, whereas the other two models are able to compute scores for all pairs in the test set. In addition, Erk et al.'s model is much more expensive to compute. Our model computes selectional preference scores for the test set in a matter of seconds, whereas for Erk et al.'s model, we ended up sampling from the test set, as computing preference values for the complete test set proved prohibitively expensive.

4.3.2 Three-way model

Table 4 compares the results of our neural network architecture for three-way selectional preference acquisition to the results of the tensor-based factorization method (Van de Cruys, 2009).

model	accuracy ($\mu \pm \sigma$)
Van de Cruys (2009)	.874 ± .001
3-way neural network	**.889** ± .001

Table 4: Comparison of model results for three-way selectional preference acquisition – mean accuracy and standard deviations of 10-fold cross-validation results

The results indicate that the neural network approach slightly outperforms the tensor-based factorization method. Again the model difference is sta-

[2] Our own implementation of Rooth et al.'s (1999) algorithm is based on non-negative matrix factorization (Lee and Seung, 2000). Non-negative matrix factorization with Kullback-Leibler divergence has been shown to minimize the same objective function as EM (Li and Ding, 2006).

[3] The best scoring model presented by Van de Cruys (2009) also uses 300 latent factors; using more factors does not improve the results.

tistically significant (paired t-test, $p < 0.01$). Using our adapted training objective, the neural network is clearly able to learn a rich model of three-way selectional preferences, reaching state of the art performance.

4.4 Examples

We conclude our results section by briefly presenting a number of examples that illustrate the kind of semantics present in our models. Similar to neural language models, the predicate and argument embedding matrices of our neural network contain distributed word representations, that capture the similarity of predicates and arguments to other words.

Tables 5 and 6 contain a number of nearest neighbour similarity examples for predicate and arguments from our two-way neural network model. The nearest neighbours were calculated using standard cosine similarity.

DRINK	PROGRAM	INTERVIEW	FLOOD
SIP	RECOMPILE	RECRUIT	INUNDATE
BREW	UNDELETE	PERSUADE	RAVAGE
MINCE	CODE	INSTRUCT	SUBMERGE
FRY	IMPORT	PESTER	COLONIZE

Table 5: Nearest neighbours of 4 verbs, calculated using the distributed word representations of embedding matrix **V** from our two-way neural network model

Table 5 indicates that the network is effectively able to capture a semantics for verbs. The first column – verbs similar to DRINK – all have to do with food consumption. The second column contains verbs related to computer programming. The third column is related to human communication; and the fourth column seems to illustrate the network's comprehension of FLOOD having to do with invasion and water.

PAPER	RASPBERRY	SECRETARY	DESIGNER
BOOK	COURGETTE	PRESIDENT	PLANNER
JOURNAL	LATTE	MANAGER	PAINTER
ARTICLE	LEMONADE	POLICE	SPECIALIST
CODE	OATMEAL	EDITOR	SPEAKER

Table 6: Nearest neighbours of 4 direct objects, calculated using the distributed word representations of embedding matrix **O** from our two way neural network model

Similarly, table 6 shows the network's ability to capture the meaning of nouns that appear as direct objects to the verbs. Column one contains things that can be read. Column two contains things that can be consumed. Column three seems to hint at supervising professions, while column four seems to capture creative professions.

A similar kind of semantics is present in the embedding matrices of the three-way neural network model. Tables 7, 8, and 9 again illustrate this using word similarity calculations.

SEARCH	DIMINISH	CONFIGURE	PROSECUTE
CLICK	LESSEN	AUTOMATE	CRITICISE
BROWSE	DISTORT	SCROLL	URGE
SCROLL	HEIGHTEN	PROGRAM	DEPLORE
UPLOAD	DEGRADE	INSTALL	CONDEMN

Table 7: Nearest neighbours of 4 verbs, calculated using the distributed word representations of embedding matrix **V** from our three-way neural network model

Table 7 shows the network's verb semantics for the three-way case. The first column is related to internet usage, the second column contains verbs of scalar change, column three is again related to computer usage, and column four seems to capture 'mending' verbs.

FLOWER	COLLEGE	PRESIDENT	SONG
FISH	UNIVERSITY	BUSH	FILM
BIRD	INSTITUTE	BLAIR	ALBUM
SUN	DEPARTMENT	MP	PLAY
TREE	CENTRE	CHAIRMAN	MUSIC

Table 8: Nearest neighbours of 4 subjects, calculated using the distributed word representations of embedding matrix **S** from our three way neural network model

Table 8 illustrates the semantics for the subject slot of our three-way model. The first column captures nature terms, the second column contains university-related terms, the third column contains politicians/government terms, and the fourth column contains art expressions.

Finally, table 9 demonstrates the semantics of our three-way model's object slot. Column one generally contains housing terms, column two contains various locations, column three contains dining occasions, and column four contains textual expressions.

WALL	PARK	LUNCH	THESIS
FLOOR	STUDIO	DINNER	QUESTIONNAIRE
CEILING	VILLAGE	MEAL	DISSERTATION
ROOF	HALL	BUFFET	PERIODICAL
METRE	MUSEUM	BREAKFAST	DISCOURSE

Table 9: Nearest neighbours of 4 direct objects, calculated using the distributed word representations of embedding matrix **O** from our three way neural network model

Note that the embeddings for the subject and the object slot is different, although they mostly contain the same words. This allows the model to capture specific semantic characteristics for words given their argument position. *Virus*, for example, is in subject position more similar to active words like *animal*, whereas in object position, it is more similar to passive words like *cell*, *device*. Similarly, *mouse* in subject position tends to be similar to words like *animal*, *rat* whereas in object position it is similar to words like *web*, *browser*.

These examples, although anecdotal, illustrate that our neural network model is able to capture a rich semantics for predicates and arguments, which subsequently allows the network to make accurate predictions with regard to selectional preference.

5 Conclusion and future work

In this paper, we presented a neural network approach to the acquisition of selectional preferences. Inspired by recent work on neural language models, we proposed a neural network model that learns to discriminate between felicitous and infelicitous arguments for a particular predicate. The model is entirely unsupervised, as preferences are learned from unannotated corpus data. Positive training instances are constructed from attested corpus data, while negative instances are constructed from randomly corrupted instances. Using designated network architectures, we are able to handle standard two-way selectional preferences as well as multi-way selectional preferences. A quantitative evaluation on a pseudo-disambiguation task shows that our models achieve state of the art performance. The results for our two-way neural network are on a par with Erk et al.'s (2010) similarity-based approach, while our three-way neural network slightly outperforms the tensor-based factorization model (Van de Cruys, 2009) for multi-way selectional preference induction.

We conclude with a number of issues for future work. First of all, we would like to investigate how our neural network approach might be improved by incorporating information from other sources. In particular, we think of initializing our embedding matrices with distributed representations that come from a large-scale neural language model (Mikolov et al., 2013). We also want to further investigate the advantages and disadvantages of having different embedding matrices for different argument positions in our multi-way neural network. In our results section, we demonstrated that such an approach allows for more flexibility, but it also adds a certain level of redundancy. We want to investigate the benefit of our approach, compared to a model that shares the distributed word representation among different argument positions. Finally, we want to investigate more advanced neural network architectures for the acquisition of selectional preferences. In particular, neural tensor networks (Yu et al., 2013) have recently demonstrated impressive results in related fields like speech recognition, and might provide the necessary machinery to model multi-way selectional preferences in a more profound way.

References

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Shane Bergsma, Dekang Lin, and Randy Goebel. 2008. Discriminative learning of selectional preference from unlabeled text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 59–68. Association for Computational Linguistics.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information & lexicography. *Computational Linguistics*, 16(1):22–29.

Stephen Clark and David Weir. 2001. Class-based probability estimation using a semantic hierarchy. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 95–102. Association for Computational Linguistics.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Katrin Erk, Sebastian Padó, and Ulrike Padó. 2010. A flexible, corpus-driven model of regular and inverse selectional preferences. *Computational Linguistics*, 36(4):723–763.

Katrin Erk. 2007. A simple, similarity-based model for selectional preferences. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 216–223, Prague, Czech Republic, June. Association for Computational Linguistics.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational linguistics*, 28(3):245–288.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Daniel D. Lee and H. Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*, pages 556–562.

Hang Li and Naoki Abe. 1998. Generalizing case frames using a thesaurus and the MDL principle. *Computational linguistics*, 24(2):217–244.

Tao Li and Chris Ding. 2006. The relationships among various nonnegative matrix factorization methods for clustering. In *Data Mining, 2006. ICDM'06. Sixth International Conference on*, pages 362–371. IEEE.

Dong C. Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1-3):503–528.

Diana McCarthy and John Carroll. 2003. Disambiguating nouns, verbs, and adjectives using automatically acquired selectional preferences. *Computational Linguistics*, 29(4):639–654.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR 2013*.

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. Maltparser: A data-driven parser-generator for dependency parsing. In *Proceedings of LREC-2006*, pages 2216–2219.

Diarmuid Ó Séaghdha and Anna Korhonen. 2012. Modelling selectional preferences in a lexical hierarchy. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 170–179. Association for Computational Linguistics.

Diarmuid Ó Séaghdha. 2010. Latent variable models of selectional preference. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 435–444. Association for Computational Linguistics.

Sebastian Padó, Ulrike Padó, and Katrin Erk. 2007. Flexible, corpus-based modelling of human plausibility judgements. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 400–409, Prague, Czech Republic, June. Association for Computational Linguistics.

Philip Resnik. 1996. Selectional constraints: An information-theoretic model and its computational realization. *Cognition*, 61:127–159, November.

Alan Ritter, Mausam, and Oren Etzioni. 2010. A latent dirichlet allocation method for selectional preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, Uppsala, Sweden, July. Association for Computational Linguistics.

Mats Rooth, Stefan Riezler, Detlef Prescher, Glenn Carroll, and Franz Beil. 1999. Inducing a semantically annotated lexicon via em-based clustering. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 104–111. Association for Computational Linguistics.

Ekaterina Shutova, Simone Teufel, and Anna Korhonen. 2013. Statistical metaphor processing. *Computational Linguistics*, 39(2):301–353.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL 2003*, pages 252–259.

Masashi Tsubaki, Kevin Duh, Masashi Shimbo, and Yuji Matsumoto. 2013. Modeling and learning semantic co-compositionality through prototype projections and neural networks. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 130–140, Seattle, Washington, USA, October. Association for Computational Linguistics.

Tim Van de Cruys. 2009. A non-negative tensor factorization model for selectional preference induction.

In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 83–90, Athens, Greece, March. Association for Computational Linguistics.

Dong Yu, Li Deng, and Frank Seide. 2013. The deep tensor neural network with applications to large vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(2):388–396.