Reduction of Parameter Redundancy in Biaffine Classifiers with Symmetric and Circulant Weight Matrices

Tomoki Matsuno[1], **Katsuhiko Hayashi**[2,4], **Takahiro Ishihara**[1], **Hitoshi Manabe**[3], **Yuji Matsumoto**[1,4]
[1]Nara Institute of Science and Technology
[2]Osaka University [3]Works Applications
[4]RIKEN Center for Advanced Intelligence Project
[1]{matsuno.tomoki.mr1,ishihara.takahiro.in0, matsu}@is.naist.jp
[2]khayashi0201@gmail.com [3]manabe_h@worksap.co.jp

Abstract

Currently, the biaffine classifier has been attracting attention as a method to introduce an attention mechanism into the modeling of binary relations. For instance, in the field of dependency parsing, the Deep Biaffine Parser by Dozat and Manning has achieved state-of-the-art performance as a graph-based dependency parser on the English Penn Treebank and CoNLL 2017 shared task. On the other hand, it is reported that parameter redundancy in the weight matrix in biaffine classifiers, which has $O(n^2)$ parameters, results in overfiting (n is the number of dimensions). In this paper, we attempted to reduce the parameter redundancy by assuming either symmetry or circularity of weight matrices. In our experiments on the CoNLL 2017 shared task dataset, our model achieved better or comparable accuracy on most of the treebanks with more than 16% parameter reduction.

1 Introduction

Recently, methods based on attention mechanisms have been used in various fields of natural language processing (Bahdanau et al., 2014; Rush et al., 2015). In tasks which deal with **binary relations** that take two words as arguments, e.g., dependency parsing, a good number of these models have achieved high performance (Kiperwasser and Goldberg, 2016; Hashimoto et al., 2017; Dozat and Manning, 2016).

Biaffine transformation is a method to incorporate an attention mechanism into binary relations proposed by Dozat and Manning (2016) (following them, we call this method **biaffine classifier**). They achieved the state-of-the-art performance among graph-based dependency parsers for the English Penn Treebank. In addition, the state-of-the-art transition-based parser on English Penn Treebank uses a biaffine classifier to evaluate the probability distribution of a word coming into the stack at a step point (Ma et al., 2018).

While biaffine transformation has rich expressiveness in modeling binary relations, its number of parameters in the weight matrix (bilinear term) is $O(n^2)$ (where n is the number of dimensions). This redundant parameters can result in high degree of freedom of the model, thus causing overfitting especially when a large number of training samples are not available (Nickel et al., 2015).

In this paper, we attempt to reduce the redundancy by introducing the assumption of either symmetry or circularity in the weight matrix at a biaffine classifier. With either assumption, we can vectorize the matrix and reduce the space complexity to $O(n)$. Additionally, the time complexity becomes $O(n)$ in the case of symmetry and $O(n \log n)$ in the case of circularity with the fast Fourier transform. Furthermore, while the expressiveness of the model based on symmetry is restricted[1], one based on circularity is able to express asymmetry relations.

In our experiments, we imposed constraints on the biaffine classifiers of the deep biaffine parser and examined the effect on the accuracy of the model. For

[1]Denote the score of the bilinear term by $f(a, b)$ when a word pair a, b has a dependency relation. In this case, if the bilinear term has symmetry, $f(a, b) = f(b, a)$. Therefore, this is not appropriate for expression of a directed edge.

32nd Pacific Asia Conference on Language, Information and Computation
Hong Kong, 1-3 December 2018

our experiments, we used the dependency parsing dataset from the CoNLL 2017 shared task (Zeman et al., 2017). We chose four languages which have relatively rich training examples and four languages which have fewer. From our experiments, we found that the method with the circularity assumption outperformed the baseline in most of the languages.

2 Deep Biaffine Parser

Models introduced in this paper are based on the **Deep Biaffine Parser** proposed by (Dozat and Manning, 2016). They achieved the state-of-the-art accuracy on the CoNLL 2017 shared task for Universal Dependencies (Zeman et al., 2017).

This model receives a sequence of words and POS tags, and calculates the probability of an arc between each pair of words as well as a syntactic function label for each arc. For evaluation of scores, it uses Long Short-Term Memories (LSTMs), Multi-Layer Perceptrons (MLPs) and biaffine classifiers.

In the following sections, first, we explain the biaffine transformation which is the essential part of a biaffine classifier while skipping explanation about LSTM and MLP for the sake of simplicity. Then, we describe the overview of the model. It is worth noting that the structure of the model is different from that in (Dozat and Manning, 2016) in that it utilizes character level information. This is because we used an updated version of the model that was made for the shared task (Dozat et al., 2017).

2.1 Biaffine Transformation

For the dependency parsing score functions, we use the biaffine transformation shown below to model binary relations. Here, $\oplus$ stands for concatenation of vectors. The first term on the right side represents relatedness score, and the second term the score of v_i and v_j appearing independently. b is bias term.

$$g(\mathbf{v}_i, \mathbf{v}_j) = \mathbf{v}_i^{\mathrm{T}} \mathbf{A} \mathbf{v}_j + (\mathbf{v}_i \oplus \mathbf{v}_j)^{\mathrm{T}} \mathbf{b} + b \qquad (1)$$

2.2 Structure of the Model

We show the structure of this model below.

1. First, this model takes two sequences: words and POS tags. It uses a unidirectional LSTM to encode each words' character-level information into a vector. It then sums this vector with a separate token-level word embeddings.

2. It then concatenates the vectors obtained above with POS embeddings and encodes them via a three layer bidirectional LSTM. y_i denotes a vector made by concatenating the hidden states (not including cell states) from the LSTMs of both directions which corresponds to the i th word w_i.

3. These outputs are then transformed with MLPs. Here, we use distinct MLPs for dependents and heads in the prediction of arcs and labels.

$$\begin{aligned} \mathbf{v}_i^{arc-head} &= \mathrm{MLP}^{arc-head}(\mathbf{y}_i), \\ \mathbf{v}_j^{arc-dep} &= \mathrm{MLP}^{arc-dep}(\mathbf{y}_j), \\ \mathbf{v}_i^{label-head} &= \mathrm{MLP}^{label-head}(\mathbf{y}_i), \\ \mathbf{v}_j^{label-dep} &= \mathrm{MLP}^{label-dep}(\mathbf{y}_j). \end{aligned} \qquad (2)$$

where $\mathbf{v}_i^{arc-head}, \mathbf{v}_j^{arc-dep} \in \mathbb{R}^n$
and $\mathbf{v}_i^{label-head}, \mathbf{v}_j^{label-dep} \in \mathbb{R}^m$

4. The scores of arcs between each word pair are calculated using a biaffine transformation.

$$s_{i,j}^{(arc)} = \mathbf{v}_i^{arc-head\mathrm{T}} \mathbf{W} \mathbf{v}_j^{arc-dep} + \mathbf{v}_i^{arc-head\mathrm{T}} \mathbf{b}^{(arc)}. \qquad (3)$$

5. Running Chu-Liu/Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967) on the scores calculated in (3), we obtain a tree structure that maximizes the total score.

6. It evaluates a score $s_{i,j}^{(l)}$ of assigning a label l ($l \in \{1, 2, ..., L\}$; L: the number of labels) on the arc between the dependent word w_i and its predicted head word w_j. The equation is defined below.

$$\begin{aligned} s_{i,j}^{(l)} &= \mathbf{v}_i^{label-head\mathrm{T}} \mathbf{U}^{[l]} \mathbf{v}_j^{label-dep} \\ &\quad + \left(\mathbf{v}_i^{label-head} \oplus \mathbf{v}_j^{label-dep}\right)^{\mathrm{T}} \mathbf{b}^{[l]} \\ &\quad + b^{[l]}. \end{aligned} \qquad (4)$$

32nd Pacific Asia Conference on Language, Information and Computation
Hong Kong, 1-3 December 2018

Here, a distinct weight matrix $\mathbf{U}^{[l]}$, weight vector $\mathbf{b}^{[l]}$ and bias $b^{[l]}$ are used for each label. The first term on the right side of the equation (4) represents the score of assigning the label l to the arc between dependent w_i and head w_j. The second term expresses the score of the label when the dependent and head are given independently.

In our experiment, $\mathbf{W} \in \mathbb{R}^{n \times n}$ and $\mathbf{U}^{[l]} \in \mathbb{R}^{m \times m}$ account for about 17% of the parameters in the deep biaffine parser. By reducing these parameters, we can expect not only improved memory efficiency but also less overfitting.

3 Proposed Methods

In this section, we introduce our two proposed methods to reduce the number of parameters of the model. We impose either a symmetry or circularity constraint on the weight matrix $\mathbf{W}$ of (3) and $\mathbf{U}^{[l]}$ ($\forall l \in \{1, 2, \ldots, L\}$) of (4).

3.1 Symmetric Matrix Constraint

This method assumes that the weight matrices of the bilinear terms are **symmetric matrices** and thus are diagonalizable. As a result, we can transform the bilinear term of the score functions into a "triple inner product" of two input vectors and a weight vector.

3.1.1 Diagonalization of the weight matrix in the bilinear term

When a matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ is symmetric, it can be diagonalized by an orthogonal matrix $\mathbf{O} \in \mathbb{R}^{n \times n}$ as below:

$$\mathbf{W} = \mathbf{O}\mathrm{diag}(\mathbf{w})\mathbf{O}^{\mathrm{T}}$$

where $\mathbf{w} \in \mathbb{R}^n$ consists of the eigenvalues of $\mathbf{W}$ and $\mathrm{diag}(\mathbf{w}) \in \mathbb{R}^{n \times n}$ represents the diagonal matrix whose diagonal elements are $\mathbf{w}$. With this property, we can rewrite the bilinear term as follows:

$$\begin{aligned} \mathbf{v}_i^{\mathrm{T}}\mathbf{W}\mathbf{v}_j &= \mathbf{v}_i^{\mathrm{T}}\mathbf{O}\,\mathrm{diag}(\mathbf{w})\mathbf{O}^{\mathrm{T}}\mathbf{v}_j \\ &= \mathbf{v}_i'^{\mathrm{T}}\,\mathrm{diag}(\mathbf{w})\mathbf{v}_j' \\ &= \langle \mathbf{v}_i', \mathbf{w}, \mathbf{v}_j' \rangle. \end{aligned} \tag{5}$$

where, $\mathbf{v}_i' = \mathbf{O}^{\mathrm{T}}\mathbf{v}_i$ and $\mathbf{v}_j' = \mathbf{O}^{\mathrm{T}}\mathbf{v}_j$, assuming $\mathbf{O}$ is learned implicitly. $\langle \mathbf{v}_i', \mathbf{w}, \mathbf{v}_j' \rangle$ is a "triple inner product" of $\mathbf{v}_i', \mathbf{w}$ and $\mathbf{v}_j'$ defined by $\langle \mathbf{a}, \mathbf{b}, \mathbf{c} \rangle = \sum_{k=1}^{n} a_k b_k c_k$. Consequently, the symmetry constraint on the matrix can reduce the number of weight parameters from n^2 to n.

3.1.2 Simultaneous Diagonalization

When a set of symmetric matrices forms a commuting family, they can be diagonalized by the same orthogonal matrix (Liu et al., 2017). So we assume the weight matrices $\mathbf{U}^{[1]}, \mathbf{U}^{[2]}, \ldots, \mathbf{U}^{[L]}$ of the scoring functions (4) form a commuting family. Namely, we assume:

$$\mathbf{U}^{[p]}\mathbf{U}^{[q]} = \mathbf{U}^{[q]}\mathbf{U}^{[p]},\ \forall p, q \in \{1, 2, \ldots, L\}.$$

With this assumption, all L weight matrices can be diagonalized simultaneously. Therefore, the vectors $\mathbf{v}_i^{label-head}$ and $\mathbf{v}_j^{label-dep}$ can be mapped by the same orthogonal matrix for all score functions.

3.1.3 Score Functions

Based on the above, under the assumption that all weight matrices are symmetric, we substitute the bilinear term in the biaffine transformations with a triple inner product. First, the score function for arc is defined as follows:

$$\begin{aligned} s_{i,j}^{(arc)} &= \langle \mathbf{v}_i^{arc-head}, \mathbf{w}, \mathbf{v}_j^{arc-dep} \rangle \\ &\quad + (\mathbf{v}_i^{arc-head} \oplus \mathbf{v}_j^{arc-dep})^{\mathrm{T}}\mathbf{b}. \end{aligned} \tag{6}$$

where, $\mathbf{w} \in \mathbb{R}^n,\ \mathbf{b} \in \mathbb{R}^{2n}$. Unlike in (3), the second term of this function contains the arc-dep vector because we confirm that it improves the performance.

Scoring functions for labels are defined as follows:

$$\begin{aligned} s_{i,j}^{(l)} &= \langle \mathbf{v}_i^{label-head}, \mathbf{u}^{[l]}, \mathbf{v}_j^{label-dep} \rangle \\ &\quad + (\mathbf{v}_i^{label-head} \oplus \mathbf{v}_j^{label-dep})^{\mathrm{T}}\mathbf{b}^{[l]}. \end{aligned} \tag{7}$$

where, $\mathbf{u}^{[l]} \in \mathbb{R}^m$, $\mathbf{b}^{[l]} \in \mathbb{R}^{2m}$. We eliminate the bias term $b^{[l]}$ in (4) because we confirm that it does not affect the performance.

3.2 Circulant Matrix Constraint

Nickel et al. (2015) used a circulant matrix for the bilinear transformation in knowledge graph completion model (Nickel et al., 2011) to reduce the number of parameters and improve the computational efficiency. Following this method, we assume the weight matrices of the bilinear term in the biaffine transformations are circulant and propose new scoring functions based on that.

3.2.1 Bilinear Transformation Using a Circulant Matrix

We define the circulant matrix $C(\mathbf{w}) \in \mathbb{R}^{n \times n}$ for a vector $\mathbf{w} \in \mathbb{R}^n$ as follows:

$$C(\mathbf{w}) = \begin{bmatrix} w_1 & w_n & \dots & w_3 & w_2 \\ w_2 & w_1 & w_n & & w_3 \\ \vdots & w_2 & w_1 & \ddots & \vdots \\ w_{n-1} & & \ddots & \ddots & w_n \\ w_n & w_{n-1} & \dots & w_2 & w_1 \end{bmatrix}. \quad (8)$$

where, $\mathbf{w}^{\mathrm{T}} = (w_1, \dots, w_n)$. . Then, we replace the bilinear term with one where the weight matrix is a circulant matrix $C(w)$ with n parameters:

$$\mathbf{v}_i^{\mathrm{T}} C(\mathbf{w}) \mathbf{v}_j. \quad (9)$$

3.2.2 Score Functions

We propose score functions that employ (9) as the bilinear term in the biaffine transformation. The score function for an arc is then defined as follows:

$$\begin{aligned} s_{i,j}^{(arc)} &= \mathbf{v}_i^{arc-head} C(\mathbf{w}) \mathbf{v}_j^{arc-dep} \\ &\quad + (\mathbf{v}_i^{arc-head} \oplus \mathbf{v}_j^{arc-dep})^{\mathrm{T}} \mathbf{b}. \end{aligned} \quad (10)$$

where, $\mathbf{w} \in \mathbb{R}^n,\ \mathbf{b} \in \mathbb{R}^{2n}$.

The score functions for labels are defined as follows:

$$\begin{aligned} s_{i,j}^{(l)} &= \mathbf{v}_i^{label-head} C(\mathbf{u}^{[l]}) \mathbf{v}_j^{label-dep} \\ &\quad + (\mathbf{v}_i^{label-head} \oplus \mathbf{v}_j^{label-dep})^{\mathrm{T}} \mathbf{b}^{[l]}. \end{aligned} \quad (11)$$

where, $\mathbf{u}^{[l]} \in \mathbb{R}^m$, $\mathbf{b}^{[l]} \in \mathbb{R}^{2m}$.

3.2.3 Efficient Computation Using Fast Fourier Transformation

In this section, we explain how to compute (9) efficiently using a fast Fourier transformation (FFT). We denote an n-point discrete Fourier transformation (DFT) matrix as $\mathfrak{F}_n \in \mathbb{C}^{n \times n}$. Then, any circulant matrix $C(\mathbf{w}) \in \mathbb{R}^{n \times n}$ can be diagonalized as follows (Gray et al., 2006):

$$C(\mathbf{w}) = \mathfrak{F}_n^{-1} \operatorname{diag}(\mathfrak{F}_n \mathbf{w}) \mathfrak{F}_n.$$

With this property, we can rewrite (9) as a triple hermitian inner product (Liu et al., 2017):

$$\begin{aligned} \mathbf{v}_i^{\mathrm{T}} C(\mathbf{w}) \mathbf{v}_j &= \mathbf{v}_i \mathfrak{F}_n^{-1} \operatorname{diag}(\mathfrak{F}_n \mathbf{w}) \mathfrak{F}_n \mathbf{v}_j \\ &= \frac{1}{n} \overline{\mathfrak{F}_n \mathbf{v}_i}^{\mathrm{T}} \operatorname{diag}(\mathfrak{F}_n \mathbf{w}) \mathfrak{F}_n \mathbf{v}_j \\ &= \langle \mathbf{v}_i', \mathbf{w}', \mathbf{v}_j' \rangle \\ &= \Re(\langle \mathbf{v}_i', \mathbf{w}', \mathbf{v}_j' \rangle). \end{aligned} \quad (12)$$

Here, $\mathbf{v}_i' = \overline{\mathfrak{F}_n \mathbf{v}_i}$, $\mathbf{v}_j' = \mathfrak{F}_n \mathbf{v}_j$, $\mathbf{w}' = \frac{1}{n} \operatorname{diag}(\mathfrak{F}_n \mathbf{w})$, and all of them are n dimensional complex vectors. $\Re(\cdot)$ is the operator which takes the real parts of its argument. With this transformation, we can compute the bilinear transformation with a circulant matrix in $O(n \log n)$ using a FFT.

The DFT of an n-dimensional vector $\mathbf{x}$, $\mathfrak{F}_n \mathbf{x}$, is conjugate symmetric if and only if $\mathbf{x}$ is a real vector (Hayashi and Shimbo, 2017). In our experiment, we initialize $\mathbf{w}'$ with the DFT of a real vector and update it in complex space. We update "frequency" domain (complex space) vectors using only the operations which have correspondence to "time domain" vectors. Thus, as described in Hayashi and Shimbo (2017), the conjugate symmetry of vectors are kept while learning because their initial values satisfy it.

3.2.4 Expressiveness of the Bilinear Transformation Using Circulant Matrices

In this section, we explain about the expressiveness of circulant matrices in relation to an arbitrary matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$. Trouillon et al. (2017) show that for any $\mathbf{W}$, there exists a normal matrix $\mathbf{W}' \in \mathbb{C}^{n \times n}$ such that $\mathbf{W} = \Re(\mathbf{W}')$. Further, as with a symmetric matrix, a normal matrix can be diagonalized as follows:

$$\mathbf{W} = \Re(\mathbf{W}') = \Re(\mathbf{O} \operatorname{diag}(\mathbf{w}') \mathbf{O}^*).$$

Here, $\mathbf{O} \in \mathbb{C}^{n \times n}$ is a unitary matrix, $\mathbf{O}^*$ is the conjugate transpose of $\mathbf{O}$ and $\mathbf{w}' \in \mathbb{C}^n$ is the complex vector which consists of the eigenvalues of $\mathbf{W}'$. The bilinear transformation whose weight matrix $\mathbf{W}$ is replaced with this, can be transformed into (12). The unitary matrix $\mathbf{O}$ is a bijective function, so the input vectors $\mathbf{v}_i$, $\mathbf{v}_j$ are learned as their one-to-one correspondent $\mathbf{v}_i' = \mathbf{O}^{\mathrm{T}} \mathbf{v}_i$, $\mathbf{v}_j' = \mathbf{O}^{\mathrm{T}} \mathbf{v}_j$, assuming that the unitary matrix $\mathbf{O}$ is learned implicitly. Note that to simultaneously diagonalize the normal matrices whose real parts are the weight matrices

$\mathbf{U}^{[1]}, \mathbf{U}^{[2]}, \ldots, \mathbf{U}^{[L]}$ in the scoring functions for labels, we have to assume that they form a commuting family as with the discussion in 3.1.2.

4 Related work

Dependency Parsing

In recent years, various graph-based parsers with attention mechanisms have been proposed.

Kiperwasser and Goldberg (2016) incorporated the attention mechanism used in machine translation (Bahdanau et al., 2014) into their graph-based parser. Their model receives vectors which are made by concatenating LSTM outputs corresponding to each word and its head candidates. Similarly, Hashimoto et al. (2017) proposed a graph-based parser where they substitute the MLP-based classifier in (Kiperwasser and Goldberg, 2016) with the bilinear one in their multi-task neural model, although they still use the MLP-based one in prediction of labels. Accordingly, Dozat and Manning (2016) modified the model by Kiperwasser and Goldberg (2016) using a biaffine classifier instead of an MLP-based one which enables the model to express not only the probability of a word receiving a particular word as dependent but also the prior probability of a word being a head.

Likewise, in transition-based parsing literature, the state-of-the-art parser on the English Penn TreeBank by Ma et al. (2018) uses an attention mechanism based on a biaffne classifier which calculates the probability distribution of the next word which comes into the stack at each time step, with LSTM outputs corresponding to each word in the input sentence. The models proposed in this paper can be incorporated into these models.

Parameter Reduction in Neural Networks

Recently, numerous methods toward parameter reduction of neural networks have been proposed.

As a similar approach to proposed methods, there is a method where a projection matrix is decomposed into smaller matrices by lower-rank approximation (Lu et al., 2016). In addition, Ishihara et al. (2018) introduced eigenvector decomposition into neural tensor networks (Socher et al., 2013) and analyzed the effects of parameter reduction. Although the paper (Ishihara et al., 2018) is similar to the present paper in that their methods address parameter reduction in the bilinear term, our work is different in that we apply it to deep biaffine parser.

There are some methods to reduce parameters in a projection matrix by sharing them. Cheng et al. (2015) used circulant matrices in the fully connected layers. Our models are different from theirs in that we use circulant matrices in the bilinear terms. Chen et al. (2015) perform parameter reduction with a hash kernel and Sindhwani et al. (2015) with special matrices like toeplitz matrices. While these can be also used for bilinear terms, the methods based on real diagonalization and circulant matrices are superior to them in computational efficiency.

Hinton et al. (2015) proposed a model called distillation and were able to train a model which was more compact than the original one. However, it needs a lot of time for training since it needs to be trained again for distillation. Hubara et al. (2016) achieved a significant reduction of parameters by the quantization, but the reported accuracy is inferior to the original model. Theoretically, these methods can be combined with the proposed methods.

5 Experiments

5.1 Dataset and Implementation

We compared the models described above on several languages in the CoNLL 2017 shared task for Universal Dependency Parsing dataset. We chose four languages which have relatively abundant training examples: UD_Chinese, UD_Czech, UD_English and UD_German. And we also selected four languages which have fewer training examples: UD_French-ParTUT, UD_Galician-TreeGal, UD_Latin and UD_Slovenian-SST.

As a baseline model, we used the dependency parser by Timothy Dozat[2] which achieved the highest accuracy on the shared task. The structures of the proposed models are based on the baseline model; we changed only the classifier part.

We only modified two hyper-parameters: we used no pretrained embeddings and initialized word embeddings with a uniform distribution. These settings remain the same throughout all experiments unless otherwise stated.

[2]https://github.com/tdozat/Parser-v2

Treebank	Baseline		Symmetry Matrix		Circulant Matrix	
	UAS	LAS	UAS	LAS	UAS	LAS
UD_Czech	93.72	91.89	93.45(-0.27)	91.50(-0.39)	**93.87(+0.15)**	**92.01(+0.12)**
UD_German	87.57	84.27	87.21(-0.36)	83.91(-0.36)	**87.61(+0.04)**	**84.39(+0.12)**
UD_English	**91.05**	**89.42**	90.95(-0.1)	89.22(-0.2)	91.04(-0.01)	89.31(-0.11)
UD_Chinese	87.67	85.58	87.55(-0.12)	85.41(-0.17)	**87.96(+0.29)**	**85.65(+0.07)**
UD_Slovenian-SST	75.63	69.53	74.94(-0.69)	68.58(-0.95)	**75.90(+0.27)**	**70.24(+0.71)**
UD_Latin	70.90	64.53	70.27(-0.63)	63.18(-1.35)	**72.38(+1.48)**	**66.00(+1.47)**
UD_French-ParTUT	91.82	89.78	**92.09(+0.27)**	89.94(+0.16)	91.94(+0.12)	**90.09(+0.31)**
UD_Galician-TreeGal	80.10	74.77	80.05(-0.05)	74.78 (+0.01)	**80.24(+0.14)**	**75.68(+0.91)**

Table 1: Main results on CoNLL 2017 dataset.

Reduced Samples	Baseline		Symmetry Matrix		Circulant Matrix	
	UAS	LAS	UAS	LAS	UAS	LAS
0 / 4	**91.05**	**89.42**	90.95(-0.1)	89.22(-0.2)	91.04(-0.01)	89.31(-0.11)
1 / 4	**90.32**	**88.57**	90.05(-0.27)	88.30(-0.27)	90.29(-0.03)	88.49(-0.08)
2 / 4	88.98	87.08	**89.15(+0.17)**	**87.17(+0.09)**	88.72(-0.26)	86.63(-0.45)
3 / 4	87.24	85.08	87.27(+0.03)	85.06(-0.02)	**87.59(+0.35)**	**85.38(+0.3)**

Table 2: Results in UD_English with fewer training samples.

Treebank	UAS	LAS
UD_Slovenian-SST	74.98(-0.65)	69.07(-0.46)
UD_Latin	71.96(+1.06)	65.68(+1.15)
UD_French-ParTUT	92.17(+0.35)	90.44(+0.66)
UD_Galician-TreeGal	79.68(-0.42)	74.82(+0.05)

Table 3: Baseline model with reduced dimensions. The numbers in parentheses are the differences from the baseline model with full dimensions.

We use gold word segmentation and gold POS tags while word segmentation and POS tagging are included in the shared task. We excluded these two tasks because the objective of this research is to show the effects of the proposed methods on biaffine classifiers which are not used for those tasks.

5.2 Results

We show the results of the baseline and two proposed methods in Table 1. The method based on circulant matrices outperformed the others on almost all languages except for English where the baseline model achieved the best accuracy and French-ParTUT where the method based on symmetric matrix did so in UAS. Interestingly, the method based on symmetric matrices underperformed the baseline on most languages. This might be because of the restricted expressiveness of a symmetric weight matrix in comparison to a circulant one especially in that the former is not appropriate for expressing asymmetric relations.

6 Analysis

In this section, we examine the robustness to overfitting of the proposed methods.

6.1 Relaxation of Overfitting

First, to further examine how numbers of parameters affect the models, we conducted experiments on UD_English treebank reducing the number of training examples by a quarter at a time.

	Number of Parameters	Percentage
Character LSTM	241200	7.64%
Bidirectional LSTM	1927200	61.03%
Arc MLP	320800	10.16%
Label MLP	80200	2.54%
Arc Classifier	160400	5.08%
Label Classifier	377437	11.95%
Others	50400	1.60%
TOTAL	3157637	100%

Table 4: Percentage of baseline model parameters accounted for by each component.

The results of this experiment are shown in Table 2. Our methods performed better on smaller datasets where the number of training examples are less than or equal to a half of the original number of examples. This indicates that our methods not only became robust to overfitting through parameter reduction but also achieved high generalizability.

Second, we ran the baseline model with classifier dimensions reduced from 400 to 200 for the arc classifier and 100 to 50 for the label classifier and compared it with the proposed methods on languages from small treebanks. As shown in Table 3, simply reducing the number of dimensions hindered the accuracies in some languages while the method based on circulant matrix consistently outperformed the baseline model with the original number of dimensions in all of these languages from small treebanks as shown in Table 3. This result indicates the effectiveness of the proposed method on smaller datasets.

6.2 Parameter Reduction

Table 4 shows the proportion of the total parameters which each part of the baseline model account for. While LSTMs occupy the largest portion of the model, the second largest part is the classifiers which account for about 17% of the parameters. Table 5 indicates that the proposed methods were able to reduce the number of parameters by more than 16%.

6.3 Parsing Speed

To test the parsing speed, we used an NVidia GTX1080 GPU and parsed the test dataset of UD_English. As mentioned in Section 3, both proposed methods are superior to the baseline model in terms of time complexity. Actually, the methods based on symmetry took 13.86 seconds, circularity 15.06 and the baseline 17.76 seconds. These results are in accordance with theoretical time complexity.

7 Conclusion

In this paper, we reduced the number of parameters in the weight matrices in biaffine classifiers based on the assumption of symmetry or circularity and examined their effects on the CoNLL 2017 shared task for Universal Dependency Parsing dataset. As a result, the method based on circulant matrices outperformed the baseline model in most of languages with about 16% parameter reduction. As future work, the L1 regularization method for CompLex (Trouillon et al., 2016) proposed by (Manabe et al., 2018) may be integrated into our methods to further reduce the number of parameters in the bilinear function. The script for the bilinear functions proposed here is provided on the first authors Github page [3].

8 Acknowledgments

We are grateful to Michael Wentao Li for proofreading the present paper. We thank the anonymous reviewers. This work was supported by JSPS KAKENHI Grant Number JP18K11457 and JST CREST Grant Number JPMJCR1513, Japan.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. URL `http://arxiv.org/abs/1409.0473`.

Wenlin Chen, James Wilson, Stephen Tyree, Kilian Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In *International Conference on Machine Learning*, pages 2285–2294, 2015.

Yu Cheng, Felix X. Yu, Rogério Schmidt Feris, Sanjiv Kumar, Alok N. Choudhary, and Shih-Fu Chang. An exploration of parameter redundancy in deep networks with circulant projections. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2857–2865, 2015. doi: 10.1109/ICCV.2015.327. URL `https://doi.org/10.1109/ICCV.2015.327`.

Chu and Liu. On the shortest arborescence of a directed graph. *Science Sinica*, Vol.14, 1965.

[3] https://github.com/TomokiMatsuno/PACLIC32/blob/master/my_linalg.py

	Baseline	Symmetry Matrix	Circulant Matrix
Arc Classifier	160400	1200	1600
Label Classifier	377437	11100	14800
Sum with shared parts	3157637	2632100	2636200
Difference from the baseline	0.0%	**-16.64%**	**-16.51%**

Table 5: Comparison of parameter sizes.

Timothy Dozat and Christopher D. Manning. Deep biaffine attention for neural dependency parsing. *CoRR*, abs/1611.01734, 2016. URL http://arxiv.org/abs/1611.01734.

Timothy Dozat, Peng Qi, and Christopher D Manning. Stanford's graph-based neural dependency parser at the conll 2017 shared task. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, 2017.

Jack Edmonds. Optimum branchings. *JOURNAL OF RESEARCH of the National Bureau of Standards*, 71(4), 1967.

Robert M Gray et al. Toeplitz and circulant matrices: A review. *Foundations and Trends® in Communications and Information Theory*, 2(3):155–239, 2006.

Kazuma Hashimoto, caiming xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple nlp tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933. Association for Computational Linguistics, 2017. URL http://aclweb.org/anthology/D17-1206.

Katsuhiko Hayashi and Masashi Shimbo. On the equivalence of holographic and complex embeddings for link prediction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pages 554–559, 2017. doi: 10.18653/v1/P17-2088. URL https://doi.org/10.18653/v1/P17-2088.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations constrained to +1 or -1. *CoRR*, abs/1609.07061, 2016. URL http://arxiv.org/abs/1609.07061.

Takahiro Ishihara, Katsuhiko Hayashi, Hitoshi Manabe, Masashi Shimbo, and Masaaki Nagata. Neural tensor networks with diagonal slice matrices. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 506–515, 2018. URL https://aclanthology.info/papers/N18-1047/n18-1047.

Eliyahu Kiperwasser and Yoav Goldberg. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327, 2016. URL http://aclweb.org/anthology/Q16-1023.

Hanxiao Liu, Yuexin Wu, and Yiming Yang. Analogical inference for multi-relational embeddings. *CoRR*, abs/1705.02426, 2017. URL http://arxiv.org/abs/1705.02426.

Zhiyun Lu, Vikas Sindhwani, and Tara N Sainath. Learning compact recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 5960–5964. IEEE, 2016.

X. Ma, Z. Hu, J. Liu, N. Peng, G. Neubig, and E. Hovy. Stack-Pointer Networks for Dependency Parsing. *ArXiv e-prints*, May 2018.

Hitoshi Manabe, Katsuhiko Hayashi, and Masashi Shimbo. Data-dependent learning of symmetric/antisymmetric relations for knowledge base completion. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16211.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *ICML*, volume 11, pages 809–816, 2011.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. Holographic embeddings of knowledge graphs. *CoRR*, abs/1510.04935, 2015. URL http://arxiv.org/abs/1510.04935.

Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

Vikas Sindhwani, Tara N. Sainath, and Sanjiv Kumar. Structured transforms for small-footprint deep learning. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3088–3096, 2015.

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings*

of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States., pages 926–934, 2013.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 2071–2080. JMLR.org, 2016. URL http://dl.acm.org/citation.cfm?id=3045390.3045609.

Théo Trouillon, Christopher R. Dance, Éric Gaussier, Johannes Welbl, Sebastian Riedel, and Guillaume Bouchard. Knowledge graph completion via complex tensor factorization. *Journal of Machine Learning Research*, 18:130:1–130:38, 2017. URL http://jmlr.org/papers/v18/papers/v18/16-563.html.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajic, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinkova, Jan Hajic jr., Jaroslava Hlavacova, Václava Kettnerová, Zdenka Uresova, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria dePaiva, Kira Droganova, Héctor Martínez Alonso, Çağrı Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonca, Tatiana Lando, Rattima Nitisaroj, and Josie Li. Conll 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19. Association for Computational Linguistics, 2017. doi: 10.18653/v1/K17-3001. URL http://www.aclweb.org/anthology/K17-3001.