Chinese Journal of Electronics
Vol.32, No.3, May 2023

# A Semi-shared Hierarchical Joint Model for Sequence Labeling

LIU Gongshen[1], DU Wei[1], ZHOU Jie[1], LI Jing[2], and CHENG Jie[2]

(1. *School of Cyber Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China*)

(2. *State Grid Information & Telecommunication Branch, Beijing 100761, China*)

**Abstract — Multi-task learning is an essential yet practical mechanism for improving overall performance in various machine learning fields. Owing to the linguistic hierarchy, the hierarchical joint model is a common architecture used in natural language processing. However, in the state-of-the-art hierarchical joint models, higher-level tasks only share bottom layers or latent representations with lower-level tasks thus ignoring correlations between tasks at different levels, i.e., lower-level tasks cannot be instructed by the higher features. This paper investigates how to advance the correlations among various tasks supervised at different layers in an end-to-end hierarchical joint learning model. We propose a semi-shared hierarchical model that contains cross-layer shared modules and layer-specific modules. To fully leverage the mutual information between various tasks at different levels, we design four different dataflows of latent representations between the shared and layer-specific modules. Extensive experiments on CTB-7 and CONLL-2009 show that our semi-shared approach outperforms basic hierarchical joint models on sequence tagging while having much fewer parameters. It inspires us that the proper implementation of the cross-layer sharing mechanism and residual shortcuts is promising to improve the performance of hierarchical joint natural language processing models while reducing the model complexity.**

**Key words — Multi-task learning, Neural networks, Natural language processing, Cross-layer sharing, Lexical semantics, Text tagging.**

## I. Introduction

Multi-task learning (MTL) has been widely explored in computer vision [1], speech recognition [2] and natural language processing (NLP) [3]. Specifically, it has shown tremendous success in improving overall performance for NLP tasks, such as monolingual machine translation [4], text classification [5], and sequence labeling [6]. Moreover, it can be divided into the parallel sharing and hierarchical joint learning model.

The parallel sharing works share lower layers between similar tasks or various datasets of the same task in a hard-sharing or soft-sharing manner, which can enhance the performance and overcome the over-fitting problem. The former shares lower layers among all tasks [7], [8] while the latter flexibly shares parameters with constraints [5]. However, its straightforward sharing strategy might fail to capture hierarchically-related linguistic features. In the other hand, the hierarchical joint learning architecture is proved to be quite powerful in different NLP tasks [9]–[13].

In natural language processing, the given text might first be processed by the fundamental tasks, such as word segmentation, part-of-speech, dependency parsing, named entity recognition, semantic role labeling and so on. The pipeline is arranged according to the linguistic hierarchy as illustrated in Fig.1. It decomposes the given text into various tags and is widely developed in different applications, such as information retrieval, machine reading comprehension, machine translation, etc. Given to the linguistic hierarchy information, the hierarchical MTL methods could leverage the complicated linguistic knowledge based on the simple features. In a hierarchical joint model, as depicted in Fig.2, all tasks are fed into the same model but supervised at different layers. It learns various linguistic information hierarchically, such as syntactic, grammar and semantic meaning, from simple tasks to complex ones. Thus, complex tasks could directly utilize the prior knowledge from bottom layers. Sogaard *et al.* [9], Hashimoto *et al.* [10], and Sanh *et al.* [11] proposed three variants of the RNN-based hierarchical joint mod-

Manuscript Received Nov. 1, 2020; Accepted Feb. 20, 2022. This work was supported by the Joint Funds of the National Natural Science Foundation of China (U1636112).

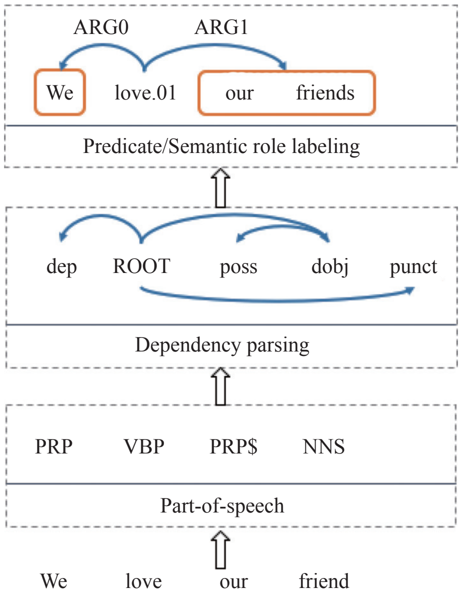

Fig. 1. Example of the linguistic hierarchical structure.

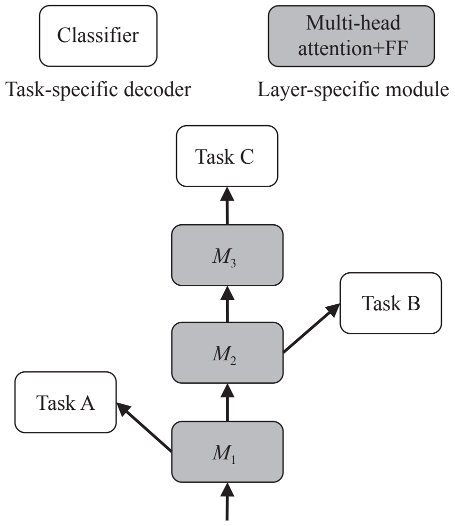

Fig. 2. The basic transformer-based hierarchical model.

els with residual latent representations and figured out the effectiveness of residual shortcuts for hierarchical sequence tagging tasks. Recently, Strubell *et al.* [6] introduced a state-of-the-art transformer-based hierarchical model that incorporates syntactic information to semantic role labeling by syntactically-informed self-attention. However, it might be difficult for the hierarchical joint model to converge when the number of tasks or the depth of model increases. Even though the hierarchical models leverage linguistic features in a bottom-up hierarchical manner, the current architecture lacks universal features among various tasks and the higher-level layers cannot be instructed by lower tasks. It is worth exploring whether hierarchical models could be further improved and shrunk.

Following the novel transformer-based hierarchical model [6], our paper investigates how to advance the correlations among hierarchical tasks with shared modules and rich latent representations in a hierarchical joint framework. We also find out an effective approach to properly merge residual shortcuts and shared modules. In order to maintain the hierarchy of linguistic features, we propose to duplicate the shared module across different layers in the original hierarchical models. Notably, the shared module is reused in a hierarchical manner during training and inference so that this shared module could be instructed by all tasks and learn how to extract universal linguistic features. Since an additional shared module is added into the hierarchical joint model, the size of those original layer-specific modules could be compacted while the performance could also be maintained or even improved. In other words, our proposed cross-layer shared module can not only advance performance but also shrink the model.

The tasks for NLP hierarchical model are usually selected according to interactions, such as semantic fundamental tasks [11], syntactic tasks [9], both syntactic and semantic tasks [6], or even fundamental and application tasks [10]. In this work, we conduct experiments across part-of-speech tagging, dependency parsing, predicate detection and semantic role labeling on CONLL-2009 Chinese subset and CTB-7.

Our contributions are as follows:

- We propose a novel semi-shared hierarchical joint model (SSHM) with a new sharing mechanism that not only shares the parameters of lower layers hierarchically, but also shares the modules across different depths of the hierarchical model parallelly.
- We explore different dataflows between the shared and layer-specific modules in the SSHM. In general, the full-combined dataflow achieves better performance compared to crossed or semi-combined dataflows. It shows that both universal and task-specific features benefit the hierarchical joint learning.
- We clarify the proper implementation of rich residual shortcuts in our semi-shared manner.
- We validate SSHM on CTB-7 and CONLL-2009 datasets. Extensive experiments show that our approach surpasses previous joint many-task models and the state-of-the-art transformer-based hierarchical joint model LISA while containing fewer parameters. Compared with LISA, our proposed model with shared heads ratio of 8/8 obtains 1.86% macro improvement with nearly 0.5 $\times$ size of parameters.

## II. Our Approach

In this section, we first introduce the transformer-based hierarchical model and residual shortcuts. Then, we clarify our semi-shared mechanism with the usage of the shared module and dataflows among shared and private modules. Also, four variants are explored via different dataflows. The overall framework for the multiple tasks are introduced at the end.

## 1. Hierarchical transformer model

As in Fig.2, the basic hierarchical transformer model is similar to the classical RNN-based hierarchical model [9]. Different tasks are supervised at different depths with a task-specific classifier. The lower tasks share all their encoder layers with higher tasks. Thus, the higher task models directly learn from the hidden states of the lower task models, which helps the deep model directly leverage the simple features from lower tasks.

Formally, suppose that the holistic transformer encoder has $J$ layers. Assume that the given sentence $X$ has $T$ tokens. It would be represented by pretrained word embedding, and then the positional embedding and word embedding are concatenated as the token inputs of the transformer encoder. The positional embedding allows the self-attention module to be aware of each token position. Then the token inputs as $\boldsymbol{H}^0$ is fed into the first layer of the transformer encoder. The transformer encoder [14] is constructed by multi-head self-attention and feed-forward network with residual connection and layer normalization. The output of the $(j-1)$-th stacked layer is taken as the input of the $j$-th one. Based on the scaled dot product attention, the multi-head attention sublayer obtains intermediate representation output $\overline{\boldsymbol{H}}^j$ by

$$\overline{\boldsymbol{H}}^j = \mathrm{LN}(\mathrm{SelfAtt}(\boldsymbol{H}^{j-1}) + \boldsymbol{H}^{j-1}) \tag{1}$$

where $\mathrm{LN}(\cdot)$ denotes the layer normalization operation. $\mathrm{SelfAtt}(\cdot)$ means the multi-head self-attention mechanism. The multi-head attention submodule contains $N$ attention heads, which enables $N$ distinct scaled self-attention transformation views to attend to the tokens in text sequence. As for each attention head, the input $\boldsymbol{H}^{j-1}$ is firstly projected into $\boldsymbol{Q}^j \in \mathbb{R}^{T\times d_k}$, $\boldsymbol{K}^j \in \mathbb{R}^{T\times d_k}$, $\boldsymbol{V}^j \in \mathbb{R}^{T\times d_v}$, respectively as query, key and value representations. Then the module multiplies $\boldsymbol{Q}^j$ by $\boldsymbol{K}^j$ and get the attention weight $\boldsymbol{A}^j \in \mathbb{R}^{T\times T}$ of the pair-wise tokens. The self-attention submodule of attention head $n$ is as follows:

$$\begin{aligned}\mathrm{SelfAtt}_n(\boldsymbol{H}^{j-1}) &= \mathrm{softmax}(\frac{\boldsymbol{Q}_n^j {\boldsymbol{K}_n^j}^T}{\sqrt{d_k}})\boldsymbol{V}_n^j \\ &= \boldsymbol{A}_n^j \boldsymbol{V}_n^j\end{aligned} \tag{2}$$

where $d_k$ is the dimension of the query and key vector, which is used to scale the dot products to avoid pushing the softmax function into regions having extremely small gradients. All outputs of $N$ attention heads is concatenated as the hidden states of the multi-head self-attention sublayer. $\overline{\boldsymbol{H}}^j$ is then fed into the feed-forward sublayer to generate the final output of this encoder layer $\boldsymbol{H}^j$:

$$\boldsymbol{H}^j = \mathrm{LN}(\mathrm{FFN}(\overline{\boldsymbol{H}}^j) + \overline{\boldsymbol{H}}^j) \tag{3}$$

where $\mathrm{FNN}(\cdot)$ represents the fully connected feed-forward networks. The feed-forward network consists of linear transformations with leaky ReLU [15], layer normalization and residual connection.

The hierarchical transformer joint model is constructed by the aforementioned stacked transformer encoder layers. And multiple tasks would be supervised at different layers with distinct classifiers as different decoders.

## 2. Residual latent representation

To leverage linguistic information better, the label embedding of lower tasks could be added into the input of higher tasks. Label representation is generated from label probabilities and label embedding. The averaged label embedding $\boldsymbol{l}_i^{(t)}$ of $i$-th token from task $t$ is calculated as

$$\boldsymbol{l}_i^{(t)} = \sum_{m=1}^{n^{(t)}} P(y_i^{(t)} = m|\boldsymbol{s}_i^{(t)}) \cdot \boldsymbol{e}_m^{(t)} \tag{4}$$

where $\boldsymbol{s}_i^{(t)}$ is the layer-normalized output, $P(y_i^{(t)} = m|\boldsymbol{s}_i^{(t)})$ is the probability of tagging as $m$ according to $\boldsymbol{s}_i^{(t)}$, $n^{(t)}$ refers to the number of labels.

$\boldsymbol{e}_m^{(t)}$ is randomly initialized label embedding of label $m$. The label representations from lower tasks and origin token embedding are concatenated as the input of higher tasks. With the layer-normalized output of task $t_{j-1}$, the input of task $t_j$ is concatenated as

$$\boldsymbol{x}_i^{(t_j)} = [\boldsymbol{s}_i^{(t_{j-1})}; \boldsymbol{e}_i^{\mathrm{w}}; \boldsymbol{l}_i^{(t_1)}; \ldots; \boldsymbol{l}_i^{(t_{j-1})}] \tag{5}$$

where $\boldsymbol{e}_i^{\mathrm{w}}$ refers to origin token embedding of the $i$-th token and $\boldsymbol{l}_i^{(t_k)}$ denotes the global representation of the $i$-th token on task $t_k$.

## 3. Semi-shared hierarchical model

The traditional hierarchical model enables complex tasks to utilize the features of simple tasks however lacks the universal features and fine-grained mutual information among all tasks. Also, the conventional architecture improve the higher tasks from the fundamental ones in a bottom-up manner but ignores the instruction of complicated tasks for the lower ones in the top-down view. In order to make multiple tasks learn from each other better and shrink the model, we propose to share parameters across lower and higher tasks at different depths in the hierarchical model. Those shared parameters can be taken as the shared module across tasks and also across various depths of the hierarchical model. To be specific, those shared modules are reused in all tasks and self-cascaded across layers in the overall structure so that it would leverage the mutual information among tasks and compact the overall hierarchical framework.

As illustrated in Fig.3, we propose to add a shared module which is reused throughout the overall model in a hierarchical manner. Thus, partial parameters, the shared ones, in higher layers learn from bottom tasks while partial parameters in lower layers also take complex linguistic information of higher tasks into account. In the other words, shared modules learn the linguistic hierarchy in both top-down and bottom-up manner. The shared module reused across the model learns to leverage both simple and complicated linguistic information from bottom to top tasks. Thus, a hierarchical model with the additional shared module could extract more abundant linguistic representations. To differ from the traditional one, we call such a hierarchical model with both the cross-layer shared module and hierarchical task-specific module as a semi-shared hierarchical model.

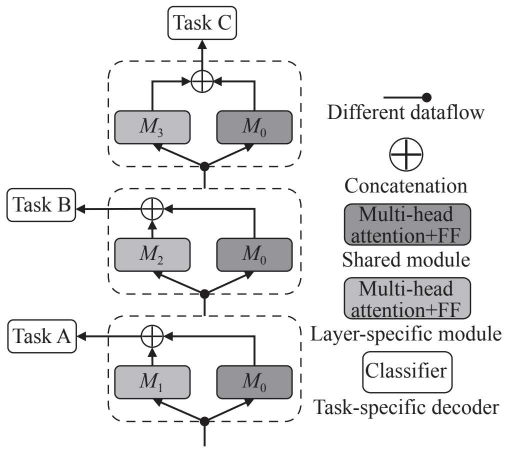

Fig. 3. The proposed semi-shared hierarchical model.

The input of task $t_j$ can be divided into the input $\boldsymbol{x}_s^{t_j}$ of the shared module $M_0$ and the input $\boldsymbol{x}_p^{t_j}$ of the layer-specific module $M_j$. Moreover, the output of task $t_j$ consists of the output $\boldsymbol{h}_s^{t_j}$ of shared module $M_0$ and the output $\boldsymbol{h}_p^{t_j}$ of layer-specific module $M_j$. The overall dataflow is as follows:

$$\begin{aligned}\boldsymbol{h}_p^{(t_j)} &= M_j(\boldsymbol{x}_p^{(t_j)})\\ \boldsymbol{h}_s^{(t_j)} &= M_0(\boldsymbol{x}_s^{(t_j)})\\ y^{(t_j)} &= \text{Decoder}([\boldsymbol{h}_p^{(t_j)};\boldsymbol{h}_s^{(t_j)}])\end{aligned} \tag{6}$$

where the final prediction $y^{(t_j)}$ of task $t_j$ is based on the concatenation of $\boldsymbol{h}_s^{t_j}$ and $\boldsymbol{h}_p^{t_j}$.

Based on the multi-head self-attention mechanism, the ratio of shared modules and layer-specific modules can be varied by the number of attention heads used in shared and layer-specific modules. The value of ratio could refer to

$$\text{Ratio} = \frac{N_s}{N_s + N_p} \tag{7}$$

where $N_s$ and $N_p$ refers to the number of attention heads in the shared and the layer-specific module. Note that the sum $N_s + N_p$ is set as a specified number while we conduct the ablation study of the sharing ratio.

The model with ratio of 0 refers to the traditional hierarchical joint model. Obviously, the parameters of our semi-shared model would decrease while the ratio increase. While the ratio equals to 1, the model is constructed by cascading the same reused module. However, the ratio would affect the performance of SSHM upon different collections of multiple tasks.

## 4. Different dataflows

The layer-specific module mainly focuses on its supervised task while the shared module pays attention to overall tasks. The former extracts task-specific linguistic information and the latter generates the common linguistic feature. Thus, the arrangement of dataflows between those two modules is critical to our semi-shared framework.

1) Straight. The simplest dataflow is to directly use their own output as the input of next task, as straight dataflow in Fig.4(a). It allows task-specific and universal features to be independent, which seems to split a hierarchical model into two parts in a parallel manner. The input of task $t_j$ is constructed as

$$\begin{aligned}x_p^{(t_j)} &= F([h_p^{(t_{j-1})}; e^w; l^{(t_1)}; \dots; l^{(t_{j-1})}])\\ x_s^{(t_j)} &= F([h_s^{(t_{j-1})}; e^w; l^{(t_1)}; \dots; l^{(t_{j-1})}])\end{aligned} \tag{8}$$

where $F$ means projection layer for linear transformation.

2) Crossed. The task-specific features might be more important than the universal information from the shared module. We could apply the cross-task module for leveraging common representations from the previous task-specific features. As Fig.4(b), the crossed variant employs the output of layer-specific module from previous task as the input for both private and shared modules. The output of shared module would not be transferred into next task but used in decoder, which allows common linguistic feature for task $t_j$ to learn from task-specific information of task $t_{j-1}$.

$$x^{(t_j)} = F([h_p^{(t_{j-1})}; e^{\text{w}}; l^{(t_1)}; \dots; l^{(t_{j-1})}]) \tag{9}$$

where $x^{(t_j)}$ is fed into both shared and task-specific modules as their inputs.

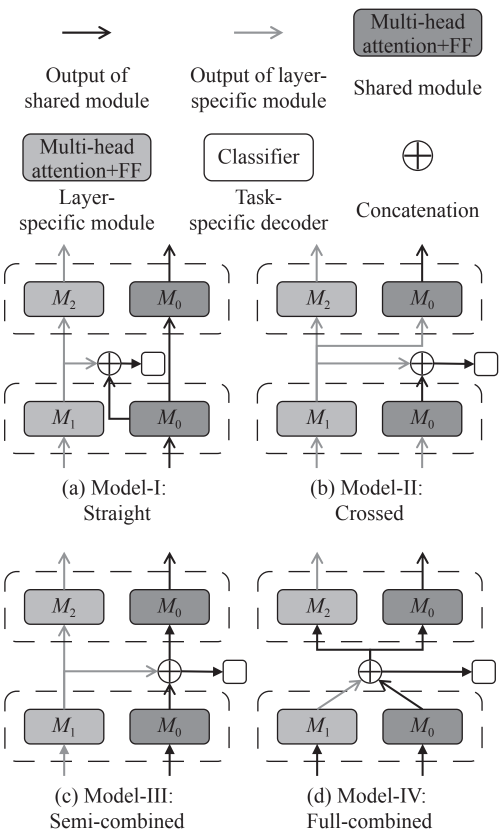

Fig. 4. Network architecture of four multi-embedding models with different data flows.

3) Semi-combined. The third variant applies the combination of task-specific outputs and universal outputs from the previous task to the shared module, which extracts common linguistic information better. In Fig.4(c), Its input consists of layer-specific input $x_p^{(t_j)}$ and shared input $x_s^{(t_j)}$, which enriches the common features.

$$x_p^{(t_j)} = F([h_p^{(t_{j-1})}; e^{\mathrm{w}}, l^{(t_1)}; \dots ; l^{(t_{j-1})}])$$
$$x_s^{(t_j)} = F([h_p^{(t_{j-1})}; h_s^{(t_{j-1})}; e^{\mathrm{w}}; l^{(t_1)}; \dots ; l^{(t_{j-1})}]) \qquad (10)$$

4) Full-combined. To extract abundant features in both private and shared modules, the input of task $t_j$ combines the outputs of those two modules from task $t_{j-1}$. As shown in Fig.4(d), the full-combined dataflow leverages advantages of semi-shared and crossed dataflow but has a slightly larger amount of parameters than the aforementioned dataflows. To obtain both task-specific and common linguistic feature better, it combines outputs of shared and layer-specific module as the input of next task.

$$x^{(t_j)} = F([h_p^{(t_{j-1})}; h_s^{(t_{j-1})}; e^{\mathrm{w}}; l^{(t_1)}; \dots ; l^{(t_{j-1})}]) \qquad (11)$$

where $x^{(t_j)}$ is the input of the shared and task-specific modules.

## 5. Multiple tasks

We conduct experiments on sequence tagging tasks, including part-of-speech tagging (POS), dependency parsing (DEP), predicate detection (PRE) and semantic role labeling (SRL). The label structure of each task is the same as that in [16]. We use Word2Vec [17] to pretrain word embedding as initial trainable token input.

1) Part-of-speech tagging

Part-of-speech tagging is a lexical task for marking each word in the given text with a specified part of speech. As for the first task, its initial inputs for both shared and layer-specific module are token embedding and position embedding. Then the concatenated output is fed into a softmax classifier to generate the probabilities of predictions. Those probabilities would be utilized in the following tasks for the global label representations of POS.

2) Dependency parsing

Dependency parsing, as a syntactic analysis task, divides the given sentence into various parts and analyze the parts with the corresponding formal grammar rules. Its input consists of the hidden state from POS, origin word embedding and label representation of POS. The input is fed into a task-specific projection layer as

$$x_{M_i}^{(\mathrm{dep})} = F_i^{(\mathrm{pos})}([\boldsymbol{h}^{(\mathrm{pos})}; e^{\mathrm{w}}; l^{(\mathrm{pos})}]) \qquad (12)$$

where projection layer differs upon the following task-specific and cross-layer shared module. Notably, the shared module utilized at POS is reused for DEP at higher layers of the overall model.

Following the syntactically self-attention mechanism [6], root token is defined as a self-loop. Head predictions are calculated by the attention weight from token $t$ to a candidate head $q$ and softmax function. Dependency labels are generated by per-class biaffine operations between the head and dependency representations [18].

3) Predicate detection

PRE is a simple task to identify the predicates in the given sentence. The predicates represents the description of the subjects in the sentence. It benefits the dependency parsing and semantic role labeling task yet strongly depends on POS. In our experiments, it directly leverage linguistic information from POS. Besides, the complexity of training a hierarchical joint learning model would increase with the depth of the overall model. Separating PRE could accelerate the learning process. Therefore, PRE directly follows POS and separates from DEP and SRL, which is different from LISA. Its input is similar to DEP, which is calculated from the task-specific projection layer with the concatenation of the hidden state from POS, origin word embedding and label representation of POS.

$$x_{M_i}^{(\text{pre})} = \boldsymbol{F}_i^{(\text{pos})}([h^{(\text{pos})}; e^{\text{w}}; l^{(\text{pos})}]) \tag{13}$$

where projection layer differs upon the following modules. The probabilities are obtained by softmax function.

4) Semantic role labeling

Semantic role labeling is a difficult semantic fundamental task in the natural language processing and assigns the semantic role of the word or phrase in the sentence. Its input consists of hidden representation of DEP, origin word embedding and label representations of POS, PRE and DEP. The input is calculated as

$$x_{M_i}^{(\text{srl})} = \boldsymbol{F}_i^{(\text{dep})}([h^{(\text{dep})}; e^{\text{w}}; l^{(\text{pos})}; l^{(\text{pre})}; l^{(\text{rel})}]) \tag{14}$$

where $l^{(\text{rel})}$ refers to dependency labels and projection layer differs upon the following module.

To make full use of syntactic information, head predictions are taken into SRL as the head attention weight. Specifically, the head attention is used as one special attention head $A_{\text{dep}}$ in multi-head self-attention submodules in the first layer of layer-specific module. Note that we use the probabilities as the attention weight $A_{\text{dep}}$ for the special attention head that replace the attention weight in (2). Thus, the self-attention submodules of the first layer in layer-specific SRL module are constructed by

$$\begin{aligned}&\text{SelfAtt}_n(\boldsymbol{H}^{j-1}) = \text{softmax}(\frac{\boldsymbol{Q}_n^j \boldsymbol{K}_n^{j\,\text{T}}}{\sqrt{d_k}})\boldsymbol{V}_n^j\\ &\text{SelfAtt}_{\text{dep}}(\boldsymbol{H}^{j-1}) = \boldsymbol{A}_{\text{dep}}^j \boldsymbol{V}_{\text{dep}}^j,\end{aligned} \tag{15}$$

where $\text{SelfAtt}_{\text{dep}}$ refers to the special attention head and $A_{\text{dep}}$ is the probability of predicted dependency head.

When it comes to the shared module, head predictions are taken as dot-product attention weight, multiplying the holistic shared input as

$$x_{M_0}^{(\text{srl})} = A_{\text{dep}} \cdot \boldsymbol{F}_0^{(\text{dep})}([h^{(\text{dep})}; e^{\text{w}}; l^{(\text{pos})}; l^{(\text{pre})}; l^{(\text{rel})}]) \tag{16}$$

where $A_{\text{dep}}$ is the probability of predicted dependency head.

The role label of token $t$ towards predicate $k$ is scored by bilinear operation of predicate-specific and role-specific representation projected from latent token representation. With the given score and transition probabilities from the training corpus, role label $y_{kt}$ is predicated through CRF layer [19], [20].

5) Training

The model is jointly trained across all tasks. The overall objective function $J(\theta)$ is defined as

$$\begin{aligned}J(\theta) = &-\frac{1}{T}\sum_{t=1}^{T}[\alpha_{\text{pos}} \log P(y_t^{(\text{pos})}|x_p^{(\text{pos})}, x_s^{(\text{pos})})\\ &+ \alpha_{\text{rel}} \log P(y_t^{(\text{rel})}|x_p^{(\text{dep})}, x_s^{(\text{dep})})\\ &+ \alpha_{\text{head}} \log P(y_t^{(\text{head})}|x_p^{(\text{dep})}, x_s^{(\text{dep})})\\ &+ \alpha_{\text{pre}} \log P(y_t^{(\text{pre})}|x_p^{(\text{pre})}, x_s^{(\text{pre})})\\ &+ \alpha_{\text{srl}} \sum_{k=1}^{T}(\log P(y_{kt}^{(\text{srl})}|x_p^{(\text{srl})}, x_s^{(\text{srl})}, y_k^{(\text{pre})})]\\ &+ \lambda\|\theta\|^2\end{aligned} \tag{17}$$

where $\theta$ is set of model parameters, $\lambda\|\theta\|^2$ is the L2-norm regularization term, $\lambda$ is a hyperparameter and $\alpha_{\text{task}}$ is penalty of different task loss.

Note that gold annotations are not leveraged during training, which is different from LISA [6]. The model is optimized by mini-batch Nadam stochastic gradient descent and gradient clipping. The learning rate schedule follows [14]. To avoid over-fitting, dropout is added into encoders and classifiers and employed in the input of label and word embedding.

## III. Experiments

We experimented our proposed model with POS, DEP, PRE and SRL on CONLL-2009 Chinese subset, using LISA[*1] [6] as our baseline model. Our model also experimented POS and DEP task on Chinese Treebank 7.0 (CTB-7). We pretrain Chinese word embedding with Wikipedia corpus[*2] by Word2Vec using Skip-gram model with negative sampling [17]. The results show that our semi-shared hierarchical joint model strongly outperforms the baseline.

### 1. Experimental setup

The CTB-7 dataset is splitted into train/dev/test subsets according to [21]. Dependencies of CTB-7 are converted by Penn2Malt tool[*3]. The head-finding rules refers to [16] and [22]. The detailed statistics is provided in Table 1. Note that the Chinese subset of CONLL-2009 is utilized in experiments.

POS and PRE are evaluated by accuracy according to the gold tag of corpus. DEP of CTB-7 corpus are evaluated excluding punctuation while DEP of CONLL-2009 are evaluated including punctuation, via standard evaluation script of "eval.pl" CONLL script. SRL is

[*1] *https://github.com/strubell/LISA*
[*2] *https://dumps.wikimedia.org/zhwiki/*
[*3] *https://cl.lingfil.uu.se/nivre/research/Penn2Malt.html*

**Table 1. Statistics of train/dev/test sets**

| Corpus | # of sentences | | |
|---|---|---|---|
| | Training set | Development set | Test set |
| CTB-7 | 31k | 10k | 10k |
| CONLL-2009 | 22k | 1.7k | 2.5k |

evaluated by standard evaluation script "eval09.pl". The results in the following tables are the accuracy of POS, labeled attachment score (LAS) and unlabeled attachment score (UAS) of DEP, labeled F1-score of SRL and macro average F1-score of joint DEP and SRL.

Hyperparameters and settings are similar to [6]. We set different regularization coefficient for classifier and transformer module, $10^{-9}$ for transformer and $10^{-8}$ for classifier. The dropout rate is set to 0.2 for residual shortcuts, token embedding and average label embedding while dropout rate for Transformer and classifiers is set to 0.1.

The shared module is designed as a transformer encoder with 3 layers. The total layer-specific module is a 11-layer transformer encoder. POS is assigned at the second layer. DEP is assigned at the fifth layer and SRL is at the end. Whereas, following POS, PRE is predicted by additional 1-layer encoder. The sum of the number of multi-heads in the shared and layer-specific modules is set to 8 and each has the dimension of 25. In the ablation study of dataflows and residual shortcuts, the ratio of shared heads is 2/8 .

## 2. Different dataflows

To evaluate the effectiveness of various dataflows, we validate these four proposed hierarchical model architectures on four tasks given above. To be consistent, the non-shared and semi-shared models used in experiments integrate word embedding of tokens and label embedding of POS tags, dependency labels as residual latent representations. Table 2 presents the performance comparison of different dataflows on CONLL-2009 development and test subset. The non-shared model is trained with residual shortcuts but without any shared module. Apart from LISA that employs gold head in training, our models are trained with predicted head and labels.

**Table 2. Performance of different dataflows**

| Development subset | POS | LAS | PRE | SRL | Macro |
|---|---|---|---|---|---|
| LISA | 92.37 | 78.36 | 95.78 | 74.33 | 76.35 |
| Non-shared | 92.96 | 77.21 | 96.39 | 75.54 | 76.38 |
| Model-I | 93.09 | 77.58 | 96.33 | 75.72 | 76.65 |
| Model-II | **93.17** | 79.13 | 96.48 | 77.08 | 78.11 |
| Model-III | 92.84 | 78.05 | 96.30 | 75.82 | 76.94 |
| Model-IV | 93.09 | **80.47** | **96.50** | **77.24** | **78.86** |
| Test subset | POS | LAS | PRE | SRL | Macro |
| LISA [6] | 92.12 | 78.89 | 95.56 | 74.51 | 76.70 |
| Non-shared | 92.82 | 77.22 | 96.25 | 74.86 | 76.04 |
| Model-I | 92.94 | 77.86 | 96.27 | 75.40 | 76.63 |
| Model-II | **93.00** | 79.26 | 96.35 | **76.57** | 77.92 |
| Model-III | 92.82 | 78.35 | 96.16 | 75.43 | 76.89 |
| Model-IV | 92.96 | **80.04** | **96.37** | 76.43 | **78.24** |

Compared to LISA and the baseline non-shared model, our semi-shared mechanism consistently results in substantial improvement on four tasks, especially for higher-level complex tasks such as DEP and SRL. Meanwhile, we find crossed dataflow (Model-II) and fully-combined dataflow model (Model-IV) outperform two other models significantly.

Specifically, Model-IV archives 76.57% F1-score on SRL with an increase of 1.17% and 1.14% points compared to Model-I and Model-III while Model-IV obtains much better results in DEP and has similar nice results of SRL. Model-IV outperforms LISA by about 0.8–1.5% points in all four tasks, especially 1.54% improvement on macro F1-score. The semi-shared mechanism has somehow shrunk the hierarchical model. To improve overall multi-tasks, Model-IV is the best method among the mentioned variants.

## 3. Residual latent representation

Apart from dataflow, residual latent representation also plays an important role in the hierarchical model [10], [11]. In what follows, we empirically investigate the effectiveness of different combinations of the residual latent representation on Model-IV (Fig.4(d)) due to its highly efficient sharing mechanism referring to Table 2. Moreover, various shortcuts also experiment on the baseline non-shared model.

Table 3 provides the ablation study of residual latent representations. Note that w/o denotes the model is trained without residual shortcuts: word (word embedding), pos (part-of-speech tag embedding), pre (predicate label embedding), dep (dependency parsing label embedding). There is large performance discrepancy after residual word embedding is deprived. Specifically, Model-IV is less susceptible to latent representations than Model-II, which even leads to 79.02% macro score even if all the latent label embedding are removed. This is because Model-IV has densely shared the useful features among multiple tasks thus decreasing the dependency on auxiliary information of label embedding. On the contrary, due to insufficient communications between different tasks, Model-II suffers from a significant performance loss without word or label embedding.

**Table 3. Experimental results with different combinations of residual latent representations**

| Model | Residual shortcuts | Development set | | | | | Test set | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | POS | LAS | PRE | SRL | Macro | POS | LAS | PRE | SRL | Macro |
| LISA [6] | w/o all | 92.37 | 78.36 | 95.78 | 74.33 | 76.35 | 92.12 | 78.89 | 95.56 | 74.51 | 76.70 |
| Non-shared | all | 92.96 | 77.21 | 96.39 | 75.54 | 76.38 | 92.82 | 77.22 | 96.25 | 74.86 | 76.04 |
| | w/o word | 93.11 | 77.33 | 96.39 | 75.53 | 76.43 | 92.79 | 77.51 | 96.13 | 74.95 | 76.23 |
| | w/o pos + pre + dep | 92.96 | 77.34 | 96.26 | 76.52 | 76.93 | 92.74 | 77.27 | 96.05 | 75.83 | 76.55 |
| Model-II | all | **93.17** | 79.13 | 96.48 | 77.08 | 78.11 | 93.00 | 79.26 | 96.35 | 76.57 | 77.92 |
| | w/o word | 92.82 | 76.20 | 96.26 | 74.64 | 74.89 | 92.59 | 76.10 | 96.04 | 73.68 | 74.89 |
| | w/o pos + pre + dep | 93.00 | 75.53 | 96.13 | 75.47 | 75.50 | 92.74 | 75.69 | 95.98 | 74.89 | 75.29 |
| Model-IV | all | 93.09 | 80.47 | **96.50** | 77.24 | 78.86 | **93.04** | 80.57 | **96.42** | 76.84 | 78.71 |
| | w/o word | 92.64 | 80.12 | 96.39 | 76.65 | 78.39 | 92.59 | 80.25 | 96.25 | 75.94 | 78.10 |
| | w/o word + pos | 92.91 | 79.15 | 96.49 | 76.13 | 77.64 | 92.78 | 79.33 | 96.34 | 75.89 | 77.61 |
| | w/o word + pre + dep | 92.51 | 77.51 | 96.03 | 75.87 | 76.69 | 92.51 | 77.88 | 96.03 | 75.59 | 76.74 |
| | w/o pos + pre + dep | 93.01 | **80.48** | 96.42 | **77.84** | **79.16** | 92.84 | 80.48 | 96.35 | **77.56** | **79.02** |
| | w/o all | 92.42 | 80.11 | 96.36 | 77.07 | 78.59 | 92.33 | 80.33 | 96.27 | 76.72 | 78.53 |

Furthermore, with semi-shared mechanism, residual latent representations do help hierarchical models obtain better results. Without training randomly initialized label embedding, SSHM with residual shortcut of only word embedding leads to the best performance. However, the model with all word and label embedding could provide pretrained label embedding of those fundamental tasks, which might benefit different NLP applications.

## 4. Shared heads ratio

The shared parameters affect the performance of our semi-shared hierarchical model over multiple tasks. Furthermore, the setting of shared ratio is critical to the performance of our SSHM. Specifically, Model-IV is evaluated with various ratio of shared heads on the CONLL-2009 Chinese subset. The experiment of POS and DEP includes residual shortcut of word and POS tag embedding. Whereas, the experiment of all four tasks only employs residual word embedding.

According to Table 4 and Fig.5, there is a slight decrease for joint learning of all four tasks while the ratio of shared heads is increasing from 2/8 to 8/8. Note that the ratio of 0/8 refers to non-shared model. Compared with LISA, the projection layers among the layer-specific and shared modules make the parameter size seems to be slightly larger than LISA in ratio from 0/8 to 2/8. As a semantic task, SRL focuses on semantic meaning while POS and DEP focus on grammar and syntax. Thus, even though SRL leverage syntactic information to improve results, performance might decrease while we try on sharing too many parameters. However, compared to LISA, our semi-shared strategy with rich residual representations could obtain great improvement of 2.5% SRL F1 and 1.86% macro F1 points

**Table 4. Results for different shared heads ratio**

| Ratio | Size of parameters | POS | LAS | UAS | PRE | SRL | Macro |
|---|---|---|---|---|---|---|---|
| LISA [6] | 1 × | 92.12 | 78.89 | 83.65 | 95.56 | 74.51 | 76.70 |
| 0/8 | 1.01 × | 92.74 | 77.27 | 81.53 | 96.05 | 75.83 | 76.55 |
| 1/8 | 1.08 × | 92.77 | **80.73** | **85.14** | **96.43** | **77.91** | **79.32** |
| 2/8 | 1.03 × | 92.84 | 80.48 | 84.78 | 96.35 | 77.56 | 79.02 |
| 4/8 | 0.96 × | 92.58 | 80.18 | 84.51 | 96.25 | 76.90 | 78.54 |
| 6/8 | 0.90 × | 92.81 | 79.75 | 84.19 | 96.23 | 76.70 | 78.23 |
| 8/8 | **0.46** × | **92.87** | 80.10 | 84.47 | 96.36 | 77.01 | 78.56 |

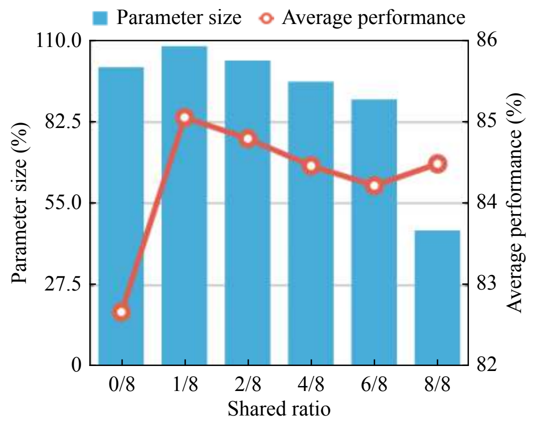

Fig. 5. The ablation study of shared ratio on all tasks.

even with only 0.46 × size of parameters.

Table 5 and Fig.6 provide the performance of Model-IV for joint syntactic tasks of POS and DEP. Parameter refers to the number of parameters without word embedding. As the ratio of shared heads increases, there will be a notable promotion for both tasks with fewer parameters. It shows that shared parameters, among different depth in hierarchical model, could not only shrink model but also obtain strong improvement of performance, in the case of strongly related tasks.

These results show that our semi-shared mechanism has promising performance in sequence tagging tasks with smaller model, which indicates its capacity of rich linguistic hierarchy information.

**Table 5. Comparison of joint POS & DEP learning**

| Ratio | Parameter | POS | LAS | UAS |
|---|---|---|---|---|
| 1/8 | 1× | 93.04 | 80.44 | 84.86 |
| 2/8 | 0.96× | 93.23 | 80.22 | 84.64 |
| 6/8 | 0.94× | 93.12 | 80.39 | 84.75 |
| 8/8 | **0.55×** | **93.42** | **80.77** | **85.10** |

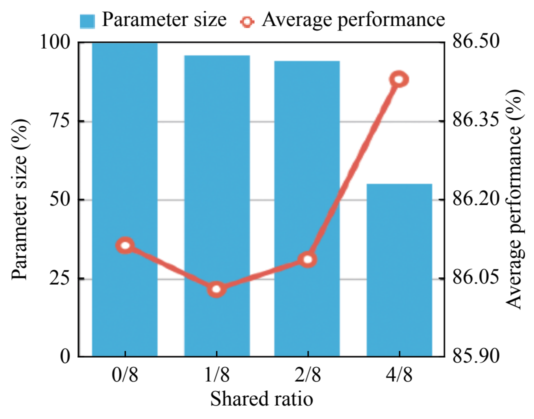

Fig. 6. The ablation study of shared ratio on POS & DEP.

## 5. Comparison

To demonstrate the performance of our semi-shared hierarchical model, we compare our models with current state-of-the-art MTL approaches on CONLL-2009 and CTB-7. Extensive experiments show that our method successfully leverages latent features of multiple tasks and results in significant performance improvement even if not using gold tags of lower tasks such as POS, PRE or DEP in higher tasks.

Table 6 shows the comprehensive comparison with previous MTL works that are jointly learning for SRL and DEP tasks. Note that we set '–' for the usage of gold tags and the Model-IV has ratio of 1/8 with only word embedding as residual shortcut. Compared to LISA, our Model-IV achieves better performance on all four tasks and leads to a higher 79.32% macro F1, even without gold tags of lower tasks as auxiliary information for higher tasks during training. It implies its potentiality of finetuning on higher tasks by data argumentation of a complex task without gold tags of lower tasks. According to Table 4, our model with only 0.46× size of LISA outperforms in all tasks and improves 1.86% marco F1 points. Our model also strongly outperforms the top 3 results in the CONLL-2009 shared tasks of joint DEP and SRL even without leveraging the gold tag of lower tasks during predicting.

**Table 6. Comparison of all four tasks**

| Model | POS | LAS | PRE | SRL | Macro |
|---|---|---|---|---|---|
| LISA [6] | 92.12 | 78.89 | 95.56 | 74.51 | 76.70 |
| Che *et al.* [23] | – | 75.49 | – | 77.15 | 76.38 |
| Zhao *et al.* [24] | – | 75.67 | – | 76.77 | 76.23 |
| Gesmundo *et al.* [25] | – | 76.11 | – | 76.05 | 76.15 |
| Ours (Model-IV) | **92.77** | **80.73** | **96.43** | **77.91** | **79.32** |

Similarly, for a board evaluation with previous works on POS & DEP, we repeat the experiment except for removing the SRL branch. For our model, the shared head ratio is 8/8.

For the model in [26], we use beam search with beam search size of 16. For the model in [27], we train it in word segmentation pipeline. As shown in Table 7 and Table 8, our semi-shared model obtains the best performance in DEP on both CTB-7 and CONLL-2009 Chinese test datasets. Our model outperforms the local model of Andor *et al.*'s [26] by 2.51% LAS and 2.88% UAS points.

**Table 7. Comparison to MTL models**

| Model | LAS | UAS |
|---|---|---|
| Ballesteros *et al.* [28] | 76.52 | 80.64 |
| Zhang & McDonald [29] | 78.57 | 82.87 |
| Lei *et al.* [30] | 76.71 | 81.67 |
| Bohnet *et al.* A [31] | 78.51 | 82.52 |
| Bohnet *et al.* B [31] | 77.00 | 81.18 |
| Albert *et al.* [32] | 79.90 | 83.57 |
| Andor *et al.* [26] B=16 | 78.26 | 82.22 |
| LISA [6] | 78.47 | 83.26 |
| Ours (Model-IV) | **80.77** | **85.10** |

**Table 8. Comparison of results on CTB-7**

| Model | POS | LAS | UAS |
|---|---|---|---|
| Yan *et al.* [27] | – | 77.93 | 81.96 |
| Ours | **92.86** | **79.13** | **82.84** |

## IV. Related Work

Deep learning methodologies have obtained tremendous success on NLP tasks. There is a tendency to improve performance of a single task by leveraging more linguistic features, which inspires lots of complex architectures.

Some approaches have made large progress in the parallel model for different tasks [33], [34]. Liu *et al.* [5] explored different methods of sharing parameters in parallel RNN models. Bert-based MT-DNN is proposed to learn representation by sharing lower layers of token embedding encoders for multiple tasks [8]. Pentyala *et al.* [35] proved the universal and task-specific features to be complementary. Ruder *et al.* [36] leverage the meta-architecture for the dynamic search of multi-task parallel sharing mechanism and focus on the latent subspace among the main task and auxiliary task. Liu *et al.* [37] proposes to learn the global features based on a single shared network and conducted task-specific feature-level attention. Standley *et al.* [38] focuses on the problem of task compatibility as it pertains to multi-task learning. Its framework offers a time-accuracy trade-off and can produce better accuracy using less inference time than both a large multi-task neural network and a single-task network.

Hierarchical model has shown its great ability of leveraging linguistic feature in the NLP tasks [9], [11], [12], [39], [40]. Some work focused on joint syntactic and semantic tasks to obtain linguistic information better, by instructing model to understand natural language with syntax [6], [10], [41]. Gong *et al.* [41] adds residual connections to the hierarchical model which jointly learns segment tagging, named entity tagging and slot filling tasks. Hashimoto *et al.* [10] proposed a joint many-task model with successive regularization method and residual latent representations, which obtained great improvement on joint tasks of fundamental sequence labeling task and semantic understanding tasks.

However, the complex successive regularization method leads to the problem of time complexity and computational complexity. Following [10], a novel work on semantic tasks also proved the effectiveness of residual token embedding [11]. Zhe *et al.* [42] learns the entity and relation extraction in a joint manner to leverage the hierarchical semantic inter-dependency.

LISA [6] shows the capacity of syntactical information on the semantic tasks, finding a new way of leveraging syntactic information by head attention [6]. Our baseline model without semi-shared mechanism is mostly similar as LISA, the state-of-the-art transformer-based hierarchical model for sequence tagging.

## V. Conclusions

We propose a new hierarchical joint model for multi-task learning, accompanied by the semi-shared strategy. Compared with previous MTL approaches, we propose to use both layer-specific and shared modules from bottom to top tasks, which is beneficial for the hierarchical model with residual latent representations. To fully investigate the latent features in multiple tasks, we design four data flows between layer-specific and shared module. Extensive experiments on CONLL-2009 Chinese subset and CTB-7 dataset demonstrate that our proposed model outperforms previous joint models and achieve great improvement on the state-of-the-art transformer-based hierarchical model. In particular, on CONLL-2009 Chinese dataset, we obtain marco F1-score of 79.32% in the overall joint experiment and LAS of 80.77% in joint POS and DEP experiment. Moreover, we explore shared ratio and different combinations of residual shortcuts and find out a simple mechanism for shrinking model and improving performance.

*The source code accompanying with this paper is available at: https://github.com/SJTUDuWei/SSHM-for-Sequence-Labeling*

## References

[1] I. Misra, A. Shrivastava, A. Gupta, *et al.*, "Cross-stitch networks for multi-task learning," in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, pp.3994–4003, 2016.

[2] L. Deng, G. Hinton, and B. Kingsbury, "New types of deep neural network learning for speech recognition and related applications: An overview," in *Proceedings of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, Canada, pp.8599–8603, 2013.

[3] R. Collobert and J. Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning," in *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, pp.160–167, 2008.

[4] D. X. Dong, H. Wu, W. He, *et al.*, "Multi-task learning for multiple language translation," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Beijing, China, pp.1723–1732, 2015.

[5] P. F. Liu, X. P. Qiu, and X. J. Huang, "Recurrent neural network for text classification with multi-task learning," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, New York, NY, USA, pp.2873–2879, 2016.

[6] E. Strubell, P. Verga, D. Andor, *et al.*, "Linguistically-informed self-attention for semantic role labeling," in *Proceedings of 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, pp.5027–5038.

[7] J. Baxter, " A Bayesian/information theoretic model of learning to learn via multiple task sampling," *Machine Learning*, vol.28, no.1, pp.7–39, 1997.

[8] X. D. Liu, P. C. He, W. Z. Chen, *et al.*, "Multi-task deep neural networks for natural language understanding," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp.4487–4496, 2019.

[9] A. Sgaard and Y. Goldberg, "Deep multi-task learning with low level tasks supervised at lower layers," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, pp.231–235, 2016.

[10] K. Hashimoto, C. M. Xiong, Y. Tsuruoka, *et al.*, "A joint many-task model: Growing a neural network for multiple NLP tasks," in *Proceedings of 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, pp.1923–1933, 2017.

[11] V. Sanh, T. Wolf, and S. Ruder, "A hierarchical multi-task approach for learning embeddings from semantic tasks," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, Honolulu, HI, USA, pp.6949–6956, 2019.

[12] J. K. Chen, K. Y. Chen, X. C. Chen, *et al.*, "Exploring shared structures and hierarchies for multiple NLP tasks," *arXiv preprint*, arXiv: 1808.07658, 2018.

[13] C. S. Gao, J. F. Zhang, W. P. Li, *et al.*, "A joint model of named entity recognition and coreference resolution based on hybrid neural network," *Acta Electronica Sinica*, vol.48, no.3, pp.442–448, 2020. (in Chinese)

[14] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach, CA, USA, pp.6000–6010, 2017.

[15] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, GA, USA, vol.28, article no.3, 2013.

[16] D. Han, P. Martínez-Gómez, Y. Miyao, *et al.*, " Effects of parsing errors on pre-reordering performance for Chinese-to-Japanese SMT," in *Proceedings of the 27th Pacific Asia Conference on Language, Information, and Computation*, Taipei, China, pp.267–276, 2013.

[17] T. Mikolov, I. Sutskever, K. Chen, *et al.*, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, Lake Tahoe, NV, USA, pp.3111–3119, 2013.

[18] T. Dozat and C. D. Manning, " Deep biaffine attention for neural dependency parsing," in *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[19] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the 18th International Conference on Machine Learning*, Williamstown, MA, USA, pp.282–289, 2001.

[20] G. M. Ling, A. P. Xu, and W. Wang, "Research of address information automatic annotation based on deep learning," *Acta Electronica Sinica*, vol.48, no.11, pp.2081–2091, 2020. (in Chinese)

[21] Y. O. Wang, J. I. Kazama, Y. Tsuruoka, *et al.*, "Improving Chinese word segmentation and POS tagging with semi-supervised methods using large auto-analyzed data," in *Proceedings of the 5th International Joint Conference on Natural Language Processing*, Chiang Mai, Thailand, pp.309–317, 2011.

[22] Y. Zhang and S. Clark, " A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing," in *Proceedings of 2008 Conference on Empirical Methods in Natural Language Processing*, Honolulu, HI, USA, pp.562–571, 2008.

[23] W. X. Che, Z. H. Li, Y. Q. Li, *et al.*, "Multilingual dependency-based syntactic and semantic parsing," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, Boulder, CO, USA, pp.49–54, 2009.

[24] H. Zhao, W. L. Chen, J. Kazama, *et al.*, " 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*, Boulder, CO, USA, pp.61–66, 2009.

[25] A. Gesmundo, J. Henderson, P. Merlo, *et al.*, "A latent variable model of synchronous syntactic-semantic parsing for multiple languages," in *Proceedings of the 13th Conference on Computational Natural Language Learning*, Boulder,

CO, USA, pp.37–42, 2014.
[26] D. Andor, C. Alberti, D. Weiss, *et al.*, "Globally normalized transition-based neural networks," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany, pp.2442–2452, 2012.
[27] H. Yan, X. P. Qiu, and X. J. Huang, "A unified model for joint Chinese word segmentation and dependency parsing," *arXiv preprint*, arXiv: 1904.04697, 2019.
[28] M. Ballesteros, C. Dyer, and N. A. Smith, "Improved transition-based parsing by modeling characters instead of words with LSTMs," in *Proceedings of 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, pp.349–359, 2015.
[29] H. Zhang and R. McDonald, "Enforcing structural diversity in cube-pruned dependency parsing," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, MD, USA, pp.656–661, 2014.
[30] T. Lei, Y. Xin, Y. Zhang, *et al.*, "Low-rank tensors for scoring dependency structures," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, MD, USA, pp.1381–1391, 2014.
[31] B. Bohnet and J. Nivre, "A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing," in *Proceedings of 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea, pp.1455–1465, 2012.
[32] C. Alberti, D. Weiss, G. Coppola, *et al.*, "Improved transition-based parsing and tagging with neural networks," in *Proceedings of 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, pp.1354–1359, 2015.
[33] X. Z. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany, pp.1064–1074, 2016.
[34] Z. H. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *arXiv preprint*, arXiv: 1508.01991, 2015.
[35] S. Pentyala, M. W. Liu, and M. Dreyer, "Multi-task networks with universe, group, and task feature learning," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp.820–830, 2019.
[36] S. Ruder, J. Bingel, I. Augenstein, *et al.*, "Latent multi-task architecture learning," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, Honolulu, HI, USA, pp.4822–4829, 2019.
[37] S. K. Liu, E. Johns, and A. J. Davison, "End-to-end multi-task learning with attention," in *Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, pp.1871–1880, 2019.
[38] T. Standley, A. Zamir, D. Chen, *et al.*, "Which tasks should be learned together in multi-task learning?," in *Proceedings of the 37th International Conference on Machine Learning*, Virtual Event, pp.9120–9132, 2020.
[39] J. M. Ye, D. X. Luo, and S. Chen, "A text error correction model based on hierarchical editing framework," *Acta Electronica Sinica*, vol.49, no.2, pp.401–407, 2021. (in Chinese)
[40] C. Feng, C. Liao, Z. R. Liu, *et al.*, "Sentiment key sentence identification based on lexical semantics and syntactic dependency," *Acta Electronica Sinica*, vol.44, no.10, pp.2471–2476, 2016. (in Chinese)
[41] Y. Gong, X. S. Luo, Y. Zhu, *et al.*, "Deep cascade multi-task learning for slot filling in online shopping assistant," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, Honolulu, HI, USA, pp.6465–6472, 2019.
[42] Z. P. Wei, Y. T. Jia, Y. Tian, *et al.*, "Joint extraction of entities and relations with a hierarchical multi-task tagging model," *arXiv preprint*, arXiv: 1908.08672, 2019.

**LIU Gongshen** received the Ph.D. degree in the Department of Computer Science, Shanghai Jiao Tong University (SJTU), China, in 2003. He is currently a Professor of SJTU. His research interests include natural language processing, machine learning, and artificial intelligent security.
(Email: lgshen@sjtu.edu.cn)

**DU Wei** received the B.E. degree from Xidian University, Xi'an, China, in 2020. He is currently working toward the Ph.D. degree with the School of Cyber Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include natural language processing and artificial intelligent security.
(Email: dddddw@sjtu.edu.cn)

**ZHOU Jie** received the M.E. degree in the School of Cyber Science and Engineering from Shanghai Jiao Tong University, Shanghai, China, in 2020. She has focused on natural language processing (NLP) since 2018. Currently, her research interests include machine learning, NLP fundamental tasks graph neural networks, and recommender system.
(Email: sanny02@sjtu.edu.cn)

**LI Jing** received the M.S. degree in computer science from Beijing University of Posts and Telecommunications, China, in 2003. She passed the Professor Level Engineer certification in State Grid. Her research interests include computer network and information security.
(Email: 1713615427@qq.com)

**CHENG Jie** received the M.S. degree in computer application technology from Beijing University of Posts and Telecommunications, China, in 2010. He joined State Grid Information and Telecommunication Branch in the same year and passed CISSP in 2020. His main research interests include enterprise-class cybersecurity, threat hunting, and XDR.
(Email: 108916685@qq.com)