Cognitive Computation (2024) 16:3020–3034
https://doi.org/10.1007/s12559-024-10320-1

RESEARCH

Granular Syntax Processing with Multi-Task and Curriculum Learning

Xulang Zhang[1] · **Rui Mao**[1] · **Erik Cambria**[1]

Received: 23 November 2023 / Accepted: 26 June 2024 / Published online: 8 July 2024

Abstract
Syntactic processing techniques are the foundation of natural language processing (NLP), supporting many downstream NLP tasks. In this paper, we conduct pair-wise multi-task learning (MTL) on syntactic tasks with different granularity, namely Sentence Boundary Detection (SBD), text chunking, and Part-of-Speech (PoS) tagging, so as to investigate the extent to which they complement each other. We propose a novel soft parameter-sharing mechanism to share local and global dependency information that is learned from both target tasks. We also propose a curriculum learning (CL) mechanism to improve MTL with non-parallel labeled data. Using non-parallel labeled data in MTL is a common practice, whereas it has not received enough attention before. For example, our employed PoS tagging data do not have text chunking labels. When learning PoS tagging and text chunking together, the proposed CL mechanism aims to select complementary samples from the two tasks to update the parameters of the MTL model in the same training batch. Such a method yields better performance and learning stability. We conclude that the fine-grained tasks can provide complementary features to coarse-grained ones, while the most coarse-grained task, SBD, provides useful information for the most fine-grained one, PoS tagging. Additionally, the text chunking task achieves state-of-the-art performance when joint learning with PoS tagging. Our analytical experiments also show the effectiveness of the proposed soft parameter-sharing and CL mechanisms.

Keywords Text chunking · Part-of-speech tagging · Sentence boundary detection · Multi-task learning · Granularity computing · Curriculum learning

Introduction

Syntactic processing is a generalization of natural language processing (NLP) subtasks that are concerned with the structure of phrases and sentences, as well as the relation of words to each other within the phrase or sentence [1]. There is a multitude in the granularity of syntactic processing. For instance, Sentence Boundary Detection (SBD), text chunking, and Part-of-Speech (PoS) tagging are all fundamental syntactic tasks, ranging from coarse-grained to fine-grained. The interplay between these tasks ensures a granular understanding of the syntactic and structural aspects of natural language, enabling more sophisticated language processing applications [2]. SBD aims to distinguish where sentences begin and end in raw texts. Downstream tasks such as machine translation [3, 4], information retrieval [5], and document summarization [6, 7] rely on predetermined sentence boundaries for good performance. In sentiment analysis, SBD can help identify negation scope to improve the performance [8]. Text chunking splits sentences into non-overlapping segments, such as noun phrase (NP) and verb phrase (VP). It helps to understand a sentence structure and the relation between words, e.g., recognizing names and syntactic components. It supports natural language processing (NLP) tasks that require a general understanding of sentence components, such as text summarization [9] and sentiment analysis [10]. PoS tagging aims to label each word in a given text with its PoS tag, e.g., noun, verb, adjective, and adverb. It parses input text to assist downstream tasks, including syntactic tasks, e.g., text chunking [11, 12] and dependency parsing [13, 14], as well as high-level NLP tasks, e.g., information retrieval [15] and sentiment analysis [16, 17], and metaphor interpretation [18–20]. These three tasks are all commonly

✉ Erik Cambria
cambria@ntu.edu.sg

Xulang Zhang
xulang001@e.ntu.edu.sg

Rui Mao
rui.mao@ntu.edu.sg

[1] College of Computing and Data Science, Nanyang Technological University, Singapore, Singapore

regarded as a sequence labeling problem. Figure 1 shows an example of the different task labels given an input sentence.

Although current works have achieved very high accuracy on these tasks [21–23], these fundamental tasks are still worth investigating for the improvement of downstream applications. On the other hand, there has been limited research on how syntactic tasks of different granularity affect each other. In traditional feature-engineering-based approaches, PoS tags are commonly used as input features for coarser-grained syntactic tasks including SBD and text chunking [24]. Modern neural-network-based techniques such as multi-task learning (MTL) [11, 25] and transfer learning [26] also show that PoS tagging is a complimentary task for text chunking, but the reverse is inconclusive.

In the age of Large Language Models (LLMs), syntactic processing techniques are often overlooked. LLMs are sub-symbolic black boxes that rely on neural networks to implicitly extract syntactic information. It can achieve human-like natural language generation capabilities. However, it does not mean that LLMs have sufficient linguistic knowledge in natural language understanding (NLU) [27, 28]. First, we believe that adopting a neurosymbolic approach, which involves decomposing tasks based on linguistic levels, could facilitate progress in NLU [2, 29]. For example, a revisiting of syntactic tasks can offer insights into understanding language structures. Second, integrating syntactic processing techniques can enhance the utility of LLMs, e.g., ChatGPT in label parsing. For instance, these techniques can facilitate the extraction of specific syntactic features, such as nominal entities and adjective-noun phrases, from predictions made by conversational LLMs. This is crucial because the conversational nature of these models often leads to the inclusion of extraneous context or explanations, which can impede users from obtaining the desired output, such as labels and entities. Lastly, the practical issue of using LLMs in syntactic processing tasks is the costs of inference. If a medium-sized model can achieve acceptable accuracy (see Tables 2, 3, and 4 for example), using LLMs for such tasks becomes a waste of computational resources. Thus, syntactic processing research is still valuable in the era of LLMs.

In this work, we conduct pair-wise MTL on SBD, text chunking, and PoS tagging respectively to study the correlations between task granularity and the complementary effect to each other. The adoption of MTL is motivated by the advantage of joint learning to mitigate the error propagation problem [33]. We propose an effective local and global dependency sharing (LGDS) mechanism. This is inspired by the finding in MTL that soft parameter-sharing allows task-specific towers to absorb useful features that are learned from their neighbor towers [34, 35].

The employed PoS tagging dataset Wall Street Journal (WSJ) [30], text chunking dataset CoNLL 2000 (CoNLL00) [36], and SBD dataset IWSLT [32] have different labels. In the case of CoNLL00, chunking, PoS, and sentence boundary labels are present. Thus, an MTL model can be trained with parallel labeled data. That is, given an input with corresponding sets of ground-truth labels for the two tasks, the model can update the parameters of the task-specific towers simultaneously. On the other hand, the WSJ dataset lacks annotated chunk labels, and the IWSLT dataset is limited to sentence boundary labels. This poses a challenge for employing MTL. For instance, in MTL for chunking and PoS tagging, the model, when presented with a WSJ training sample, cannot simultaneously update both task towers. This scenario is termed training with non-parallel labeled data in our MTL paradigm. Addressing MTL with non-parallel labeled data is significant, as optimizing a neural network-based model on input instances with non-parallel labels may introduce bias toward a specific task, potentially causing instability in the training of the other task. For example, a WSJ input instance optimizes the parameters by its associated PoS labels. As such, the neural network tends to yield PoS-tagging-efficient parameters and lower the accuracy of text chunking. MTL with non-parallel labeled data is a common MTL paradigm; however, previous research did not pay enough attention on this [37, 38].

PoS label:	PRP	VBD	PRP$	NNS	.
Chunk label:	S-NP	S-VP	B-NP	E-NP	O
Sentence Boundary label:	O	O	O	PERIOD	O
Example sentence:	I	received	her	flowers	.

Fig. 1 An example that showcases the PoS, chunk, and sentence boundary labels of a given sentence. The PoS tagging labels employ the Penn Treebank [30] annotation schema. The chunk labels employ BIOES annotation schema [31], where "B" represents the beginning of a chunk that immediately follows another chunk; "I" represents the word is inside a chunk; "O" represents outside of any chunk; "E" represents the ending word of a chunk; "S" represents a chunk phrase that contains only a single token. The sentence boundary word is labeled as PERIOD, following the tagging schema of the IWSLT dataset [32]. Note that there are no punctuation marks in the IWSLT dataset

To address this challenge, we propose to incorporate parallel labeled data to balance the biased learning on non-parallel labeled data, assuming that a strategic combination of two instances can achieve effective learning for both tasks at each batch training step. To this end, we present a Curriculum Learning (CL) mechanism that selects complementary training instances from both datasets to be packed in the same training batch. The hypothesis (H1) is that a model can achieve more robust MTL with non-parallel labeled data if the task-encoded input instances from two different datasets are in similar vector spaces. We use cross-entropy to measure the similarity between two vector spaces to select complementary samples, because cross-entropy is a classic and intuitive measure for quantifying the information difference between two probability distributions [39]. We select and train the pair of instances from two datasets in the same batch by the curriculum criterion of minimizing the cross-entropy of the hidden states from two task-specific towers.

We examine the pair-wise performance of SBD, chunking, and PoS tagging on our MTL model using three public datasets and study how syntactic tasks of different granularity affect each other. The text chunking task obtains impressive performance when jointly learned with PoS tagging, achieving state-of-the-art performance (98.43% Micro-F1), outperforming the strongest published baseline by 1.13%. The PoS tagging task and SBD task both demonstrate performance gains when MTL with each other, compared to when MTL with chunking. It may be concluded that pair-wise MTL among syntactic tasks of different granularity can bring performance gain compared to single-task learning. Nevertheless, fine-grained tasks, such as PoS tagging, tend to be more helpful for coarse-grained tasks. Whereas coarse-grained task, i.e., SBD, provides useful structural information for PoS tagging. We also conduct an ablation study to demonstrate the effectiveness of our proposed LGDS and CL mechanisms.

Our research scope does not target to achieve state-of-the-art performance for all the involved tasks. Instead, we aim to propose an MTL framework where the complementary effects between syntactic tasks of different granularity can be reliably evaluated using their respective benchmark datasets. Thus, the contribution of this work can be summarized as follows: (1) We propose an MTL framework with a novel soft parameter-sharing mechanism that shares local and global dependency information for sequence-labeling-based syntactic processing tasks; (2) we propose a CL mechanism to improve the stability of the loss convergence for MTL with non-parallel-labeled data; (3) we study how syntactic tasks with different granularity complement each other through pair-wise MTL.

Related Work

Sentence Boundary Detection

SBD is an important yet overlooked pre-processing task. It is seemingly easy to identify punctuation marks. However, the presence of a period may cause notable ambiguities, e.g., abbreviations and decimal points. Early methods focus on the disambiguation of period usage in text. Recent task definition, motivated by automatic speech recognition, becomes more challenging by aiming to classify whether a word is followed by a sentence boundary punctuation mark in unsegmented speech transcripts [32]. Rule-based approach [40–42], despite the difficulties of constructing a comprehensive enough rule set, is still employed in recent years and achieves competitive performance. Whereas deep learning approaches [43–45] are the most widely used for the SBD task nowadays. Notably, early feature engineering methods often incorporate PoS tags as a useful feature for SBD [46–48], which suggests that PoS tagging might be a complementary joint learning task for SBD.

Text Chunking

Text chunking is normally formulated as a sequence labeling task since the work of [49]. Early feature engineering methods utilized graphical models, e.g., Conditional Random Fields (CRF) [50–53].

In the era of deep learning, recent works utilized neural networks to automatically capture relevant features. The most widely used architecture is the combination of CRF and Recurrent Neural Network (RNN) variants such as Bidirectional Long Short-Term Memory (BiLSTM) [21, 54–56] and Gated Recurrent Units (GRU) [57]. To address the RNN-based method's limitation of capturing non-continuous relations between tokens, [58] proposed a Position-aware Self Attention (PSA) mechanism, where the BiLSTM encoder employs self-attention to encode relative positional information. However, there is no study on the effectiveness of learning both tasks with dependency information sharing. We believe that syntactic dependency features are useful for chunking.

Part-of-Speech Tagging

PoS tagging is a well-studied problem. Most early works, following [59], rely on hand-crafted features derived from the local N-gram context. Similar to text chunking, most feature engineering methods utilized graphical models, e.g., HMM [60, 61], Maximum Entropy Markov Model (MEMM)

[62], and CRF [63]. Among them, CRF is able to leverage decisions at different positions and compute the conditional probability of global optimal output sequence, overcoming the drawbacks of both HMM and MEMM.

In the era of deep learning, different neural methods are used to learn character- and word-level features. A window approach is proposed to extract word-level [11] and character-level [64] features from local context. [65] extended the standard BiLSTM-CRF structure with a character-level Convolutional Neural Network (CNN) layer [66]. Similarly, [21] proposed a contextual string embedding called Flair to aid the BiLSTM-CRF tagger. [67] presented a deep CNN architecture that stacked up more convolutional layers while averting the vanishing gradient problem. The accuracy of PoS tagging has been pushed to its near limit. Hence, improvement of this task should prioritize its effectiveness on aiding downstream tasks.

Multi-Task Learning

MTL takes advantage of sharing parameters and learns features from two or more tasks. It helps a machine improve performance and mitigate overfitting [68]. MTL methods that take the form of joint training can be categorized into two types, namely hard parameter-sharing and soft parameter-sharing. The former shares the hidden layers between all tasks, while using a task-specific layer for each task's output. As such, all involved tasks share the same representation space, reducing overfitting but less adept at capturing task-specific information. Soft parameter-sharing, on the other hand, keeps a separate model, i.e., task tower, for each task and utilizes constraint mechanisms to encourage similarities among task model parameters, thus more flexible for task-specific representations.

There have been studies where MTL approaches, including hard and soft parameter-sharing, are applied to multiple syntactic and semantic tasks [11, 25], e.g., PoS tagging, text chunking, named entity recognition, and semantic role labeling. MTL can also interwind with different input modalities [69] and learning paradigms [70]. However, the motivation behind the task selection of MTL stems from similar task formulation (sequence labeling), instead of linguistic granularity. As such, the tasks involved are jointly learned together indiscriminately, and their complementarity to each other is not explored.

Furthermore, there are only a few studies that effectively address the non-parallel labeled data learning issue in MTL [38]. Chen et al. [25], Liu et al. [71] proposed LSTM-based architectures that can handle non-parallel labeled data by using a shared LSTM layer between two task towers. Their limitation is that such an approach is more suitable for highly similar tasks, e.g., sentiment classification tasks in different domains or annotations. Similarly, [72] employed shared stacked Bi-LSTM-CNNs with inter-task feedback strategy to adopt hierarchical tasks for parallel multi-task learning. However, such architecture runs the risk of biasing towards one task when the volume of task datasets is imbalanced.

Curriculum Learning

CL aims to automatically select the most suitable samples for each training step [73]. The curriculum is a sequence of training criteria that rely solely on the data, the model, and the task objective. CL is widely used to select training samples from easy to difficult for efficient learning [74–76].

The signal for determining whether a sample is appropriate for the current model is considered as curriculum criterion. Traditionally, such criterion is defined as a task-dependent difficulty metric, such as input text length [75, 77] and term frequency [77, 78]. Contrasting the predefined sample ordering, another common way to dynamically generate curriculum is to use task loss as the signal for teach-student CL [79–81], or self-pace learning [74, 82–84]. However, these CL criteria are applied to enforce the easy-to-hard ordering of the curriculum. To the best of our knowledge, CL has not been used to address the issue of MTL with non-parallel labeled data.

Methodology

We conduct pair-wise MTL on three syntactic processing tasks, namely, SBD, text chunking, and PoS tagging. Our hypothesis is that syntactic tasks of varying granularity tend to display different levels of compatibility with each other. We also hypothesize that by controlling the combination of input data from different sources without parallel-annotated labels, an MTL model can achieve higher overall accuracy and smooth learning loss convergence. This is because the feature alignment of multiple task inputs can help the neural network learn features from similar spaces. In contrast, features from very different spaces may lead to unstable learning. This is particularly important for MTL in the context where the sub-tasks do not have parallel labels for the same input sentence.

In light of this, we propose an MTL framework (the first subsection) with a novel soft parameter-sharing mechanism to pass linguistic features learned from one task to the other, so as to investigate the pair-wise complementarity of the involved syntactic tasks. The soft parameter-sharing mechanism (LGDS in the second subsection) means to share local and global dependency information between two tasks. Considering that certain task datasets lack ground-truth labels for complementary tasks—for instance, the absence of PoS or chunk labels in the SBD dataset—we propose a CL mechanism (detailed in the third subsection) to enhance accuracy

and stabilize learning. This mechanism regulates the integration of input data from both tasks within the same batch.

Multi-Task Learning

We denote the two tasks involved in our MTL framework as p and c, respectively. Then, given an input sentence $w = (w_1, w_2, \ldots, w_l)$, the goal of the MTL model is to predict its task p labels $p = (p_1, p_2, \ldots, p_l)$ and its task c labels $c = (c_1, c_2, \ldots, c_l)$, where l is the sequence length.

The architecture of the model is shown in Fig. 2. The input sentence w is first embedded with pre-trained embeddings, then fed into the two respective towers for task p and task c. In each tower, the input is passed through an encoder ($Encoder_0$), whose output is denoted as H_0^p in the task p tower and H_0^c in the task c tower. In this work, we adopt Transformer [85] as the encoder, because it has been widely applied in diverse NLP tasks, presenting strong performance [86, 87].

Next, n blocks of encoders and soft parameter-sharing mechanisms ($LGDS$) are employed. Here, we denote the output of block i in the task p tower as H_i^p, and the one in the task c tower as H_i^c. Then, for the task p tower, H_i^p is given by

$$T_i^p = Encoder_i^p(H_{i-1}^p), \tag{1}$$

$$H_i^p = LGDS_i^p(T_i^p, H_{i-1}^c). \tag{2}$$

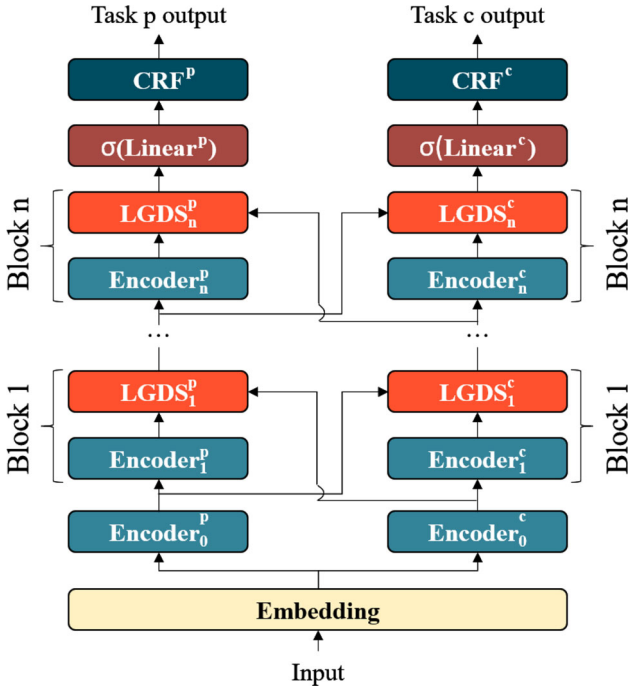

Fig. 2 Architecture of the multi-task learning framework. σ denotes a SoftMax function

For the task c tower, H_i^c is computed similarly to (2) by incorporating H_{i-1}^p through LGDS.

Next, the final hidden states H_n^p and H_n^c are each fed into a linear layer ($L_{n+1}(\cdot)$) with SoftMax (σ):

$$E^p = \sigma(L_{n+1}^p(H_n^p)), \tag{3}$$

$$E^c = \sigma(L_{n+1}^c(H_n^c)). \tag{4}$$

Finally, during training, we use E^p, E^c, CRF, and its loss function [63] to obtain losses for task p ($\mathcal{L}^p$) and task c ($\mathcal{L}^c$), respectively. The overall loss ($\mathcal{L}$) is given by

$$\mathcal{L} = \alpha\mathcal{L}^p + (1-\alpha)\mathcal{L}^c, \tag{5}$$

where α is a hyper-parameter. During inference, Viterbi decoding algorithm [88] is employed in CRF to predict the label sequences of task p and task c.

Local and Global Dependency Sharing

We propose a soft parameter-sharing mechanism named LGDS to incorporate local and global dependencies that are learned from both tasks. As shown in Fig. 3, LGDS combines CNN and Biaffine attention [13]. CNN, constricted by window size, is used to extract relevant information from the neighbor tower within the local context of the focal token. The output of the CNN in block i of the task c tower K_i^c can be computed as

$$\begin{aligned} K_i^{c'} = ReLU(&\text{Conv1D}(H_{i-1}^p, f=1) \\ &\oplus \text{Conv1D}(H_{i-1}^p, f=3) \\ &\oplus \text{Conv1D}(H_{i-1}^p, f=5)), \end{aligned} \tag{6}$$

$$K_i^c = tanh(W_k K_i^{c'} + b_k), \tag{7}$$

where $\oplus$ denotes concatenation. f denotes filter width. W_k and b_k are learnable parameters.

Biaffine attention was used in dependency parsing to capture the dependencies between each word of a sentence [13]. Thus, we use Biaffine attention to capture the long-range dependencies. A Biaffine attention matrix ($S_i^c \in \mathbb{R}^{l\times l}$) in block i of task c is computed by

$$S_i^c = T_i^c U_i^c H_{i-1}^{p\top} + H_{i-1}^p e_i^c, \tag{8}$$

where U_i^c and e_i^c are learnable parameters. The Biaffine attention output (B_i^c) is the task p information (H_{i-1}^p) enhanced by its global task-c-dependent information (S_i^c)

$$B_i^c = tanh(softmax(S_i^c)H_{i-1}^p). \tag{9}$$

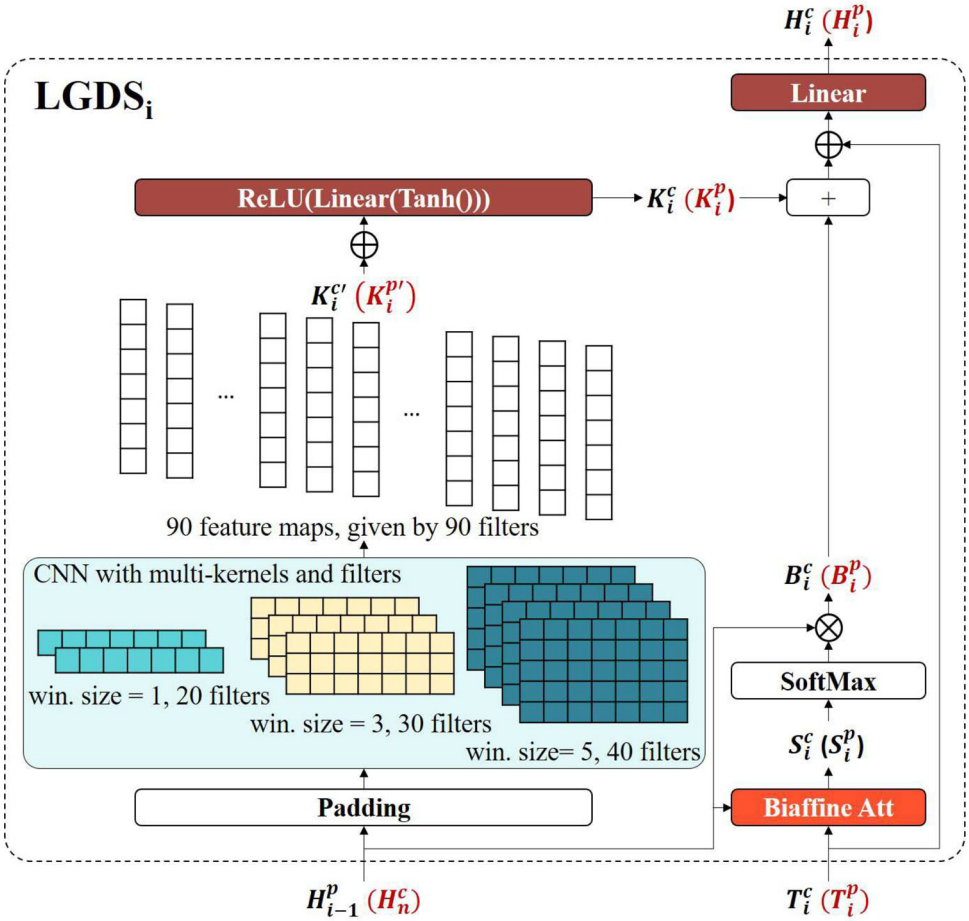

Fig. 3 Local and global dependency sharing (LGDS) mechanism. Bold italics denotes input and output variables, where black denotes the variables for learning task c, red with parentheses denotes variable for learning task p. Colored boxes denote layers with learnable parameters. + denotes a plus; ⊕ denotes concatenation; ⊗ denotes matrix multiplication

Finally, the LGDS output of the current block i from the task c tower (H_i^c) is computed as

$$H_i^c = L_i^c(T_i^c \oplus (K_i^c + B_i^c)). \tag{10}$$

The output for the LGDS in the task p tower (H_i^p) can be derived from similar procedures, while the input from the private tower is T_i^p instead of T_i^c; the input from the neighbor tower is H_{i-1}^c.

Curriculum Learning

The benchmark dataset for one task might not contain ground-truth labels for the other task, e.g., the PoS tagging dataset (WSJ) not being annotated with chunk labels, causing a non-parallel labeled data MTL issue. To alleviate this, we propose a CL mechanism to select complementary samples from the other task's dataset to optimize the training on the non-parallel-labeled task's data. Using the MTL of chunking and PoS tagging as an example, for an input WSJ instance ($w^{p,m}$), we randomly select J instances from CoNLL00 to feed into the model together for forward propagation. The length of $w^{p,m}$ is l. The CoNLL00 instances are padded or pruned to achieve the same length (l) in vector space. We denote the output of the first chunking encoder ($Encoder_0^c$) resulting from the j-th CoNLL00 instance as $T_0^{c,j}$, ($j \in \{1, \ldots, J\}$). Subsequently, we select the CoNLL00 instance whose $T_0^{c,j}$ is the most similar to $T_0^{p,m}$ (given by $w^{p,m}$ and $Encoder_0^p$) in vector space to balance the learning of $w^{p,m}$, according to cross-entropy value

$$CE^{c*} = \arg\min_j(-\sum_{k=1}^{l} T_{0,k}^{p,m} \log(T_{0,k}^{c,j})), \tag{11}$$

whose corresponding instance is defined as w^{c*}. w^{c*} and $w^{p,m}$ are learned with both forward and backward propagation in the same batch to achieve stable CL for text chunking. We use hidden states from $Encoder_0^c$ and $Encoder_0^p$ rather than task losses as signals, because minimizing losses does not allow the model to learn useful information from the current input.

Similarly, we apply this CL procedure on the SBD dataset (IWSLT) when training with PoS tagging or text chunking, as it does not contain ground-truth labels for either task. We do not apply the CL mechanism when training on parallel

annotated data, e.g., CoNLL00 with both chunking, PoS, and SBD labels, and WSJ with both PoS and SBD labels.

Experiment

Baselines

To put the performance of our MTL framework into perspective, we include the following single-task and multi-task baselines.

SBD:

- **T-BRNN** [89]: A bidirectional GRU model with attention mechanism, using GloVe embeddings.
- **BERT** [44]: A BERT-large model with Bi-LSTM-CRF stacked on top.
- **Roberta** [45]: A Roberta-large model with Bi-LSTM-CRF stacked on top.

Text chunking:

- **GRU-CRF** [57]: A deep hierarchical GRU model that encodes both character level and word level information, using fine-tuned SENNA embedding [11].
- **Star-C** [90]: A Star-Transformer-CRF model using GloVe embedding [91].
- **MVCRF** [12]: A multi-view CRF that extracts features from the word view as well as POS view, using SENNA embedding. **Flair-C** [21]: A BiLSTM-CRF model that utilizes Flair embedding, GloVe embedding, and task-trained character embedding.
- **ACE** [22]: A BiLSTM-CRF model that automatically concatenates suitable embeddings including GloVe, Flair, BERT [92], etc.

POS tagging:

- **LSTM-CNN-CRF** [65]: A BiLSTM-CRF model that uses CNN to extract character level features, using GloVe embedding and task-trained character embedding.
- **GatedDualCNN** [67]: A deep CNN architecture that employs a dual path to alleviate the vanishing gradient problem, using GloVe embedding and task-trained character embedding.
- **Star-P** [90]: Same as Star-C.
- **Flair-P** [21]: Same as Flair-C.

MTL:

- **Meta-LSTM** [25]: A MTL framework with a task-shared Meta-LSTM layer for non-parallel labeled data, using GloVe embedding.
- **Gated** [93]: A MTL model with gated network as a sharing mechanism using BERT embedding.
- **AUX** [94]: A BiLSTM-based model that concatenates the output of the auxiliary tower with input representation to feed into the primary tower.

We also conduct experiments by using BiLSTM instead of Transformer in our framework (**Ours-LSTM**) to achieve a fair comparison with other BiLSTM-CRF-based baselines.

Datasets

Our multi-task learning framework is trained and evaluated with the WSJ dataset [95], the CoNLL00 dataset [36], and the IWSLT dataset [32]. The details of the used datasets can be found in Table 1. For the SDB task, we use the PERIOD class tagging schema provided by the IWSLT dataset [32] and use the Ref testing set for evaluation. For the chunking task, we use the BIOES tagging schema [31]. For the POS tagging task, we use the Penn Treebank annotation schema [30]. The standard evaluation metrics are accuracy for POS tagging, F1 measure for text chunking, and PERIOD class F1 measure [32] for SBD, which are in line with our baselines.

Setups

For the hyper-parameters, we randomly select $J = 4$ instances for CL. We adopt $\alpha = 0.6$. We set the initial learning rate to 0.0001 and adopt a learning schedule with the step size of 20 and decay factor $\gamma = 0.5$. We also implement an early stop strategy, where the model stops training if the average accuracy of the two tasks is not improved in five epochs. We use Adam [96] with 0.9 and 0.99 betas to optimize the model. We run 30 epochs with the batch size of 20 on NVIDIA Tesla P100-PCI-E. We use Flair embedding, GloVe embedding with 300 dimensions, and task-trained character embedding as embeddings, aligning with one of the strongest baselines for two of the target tasks [21]. There are 2 blocks (n=2 in Fig. 2) of the encoder with LGDS in each task-specific

Table 1 Details of the WSJ, CoNLL 2000, and IWSLT datasets

Dataset	Task	Data	# seq.	# token
IWSLT	SBD	Train	–	2,102,417
		Dev	–	295,800
		Test	–	12,626
CoNLL 2000	Chunking	Train	8,937	211,727
		Test	2,013	47,377
WSJ	PoS tagging	Train	38,219	912,344
		Dev	5,527	131,768
		Test	5,462	129,654

tower. The Transformer-based encoders have 4 heads, and 128 dimension hidden states. Additionally, we examine BiLSTM-based encoders with 200 dimension hidden states. We report micro-F1 and accuracy for SBD, text chunking, and PoS tagging tasks, based on the averaged results of 5 runs.

Results

From Table 2, we can see that jointly learning SBD with PoS tagging outperforms that with text chunking by 1.43% F1 score. Similar performance advantages can be observed in all the MTL methods, indicating that PoS tagging is a more complementary task for SBD than chunking. Additionally, Ours-Transformer outperforms BERT in both task pair settings, and only falls behind Roberta by 1.35% F1 score when paring with PoS tagging, despite using fewer Transformer layers than both baselines. It shows that our MTL framework can pass useful features from one task tower to the other and glean the benefits of joint learning. Applying our proposed CL mechanism also consistently improve the MTL baselines Gated and AUX in both task pair settings, proving its effectiveness in bringing performance gain.

Results shown in Table 3 indicate that text chunking achieves the best performance when jointly learned with PoS tagging. Although Ours-Transformer outperforms the best single-task baseline (ACE) when learned with SBD, the extent of improvement, 0.18%, is much less significant comparing to when learned with PoS tagging, which stands at 0.83%. This contrast of complementarity can also be observed among all the experimented MTL methods. We can also see that when jointly learned with PoS tagging, Ours-LSTM outperforms the LSTM-based ACE by 0.66% in F1 scores. It shows the effectiveness of our proposed MTL task pair, soft parameter-sharing (LGDS), and CL mechanisms. Using Transformer can further improve the model, reaching 98.13% F1, achieving the best performance. It significantly outperforms Meta, showing that our approach for non-parallel labeled data in MTL is superior to existing works.

From results shown in Table 4, it can be observed that when jointly learned with SBD, all MTL methods obtains better PoS tagging performance than learned with text chunking. However, the best performing MTL method, Ours-Transformer, still lags behind the best baseline Flair-P by a marginal 0.06% in accuracy, suggesting that PoS tagging reaps limited benefits from the joint training with other syntactic tasks. Furthermore, combining with the results shown in Table 2, we can conclude that our model achieves the best PoS tagging and SBD performance when they are jointly learned, whereas seeing Table 4 with Table 3 indicate that chunking can benefit a lot from PoS tagging but not in reverse. It can be inferred that fine-grained syntactic tasks are complementary for MTL with the more coarse-grained ones, among which the most fine-grained task, namely PoS tagging, consistently contributes the most to the improvement of the other tasks. On the other hand, PoS tagging receives limited benefits from MTL with more coarse-grained tasks. Jointly learning PoS tagging with SBD achieves better performance than with chunking and comparable performance with the strongest single-task baseline. This might be due to the fact that SBD provides global sequence structure information, but is more challenging to learn in the fine-coarse processing of PoS tagging.

Combining the results in Tables 2, 3, and 4, we can further draw the conclusions that (1) when paired with the most complementary task, Our-Transformer significantly outperforms the strongest baseline in text chunking and obtains comparable performance in SBD and PoS tagging; 2) applying

Table 2 SBD results when learning with PoS tagging (SBD with PoS) and with chunking (SBD with Chunking)

Model	SBD w/ PoS		SBD w/ Chunking	
	F1	Acc	F1	Acc
T-BRNN	72.9	–	–	–
BERT	84.1	–	–	–
Roberta	**88.6**	–	–	–
Gated[a]	81.31	81.42	79.22	79.53
Gated w/CL[a]	81.58	81.64	79.69	79.88
AUX[a]	79.99	80.28	78.93	79.41
AUX w/CL[a]	80.46	80.61	79.18	79.52
Ours-LSTM	$86.74_{0.14}$	$\underline{87.01}_{0.19}$[b]	$84.43_{0.15}$	$84.86_{0.15}$[b]
Ours-Transformer	$\underline{87.25}_{0.11}$	$\mathbf{87.48}_{0.10}$[b]	$85.82_{0.16}$	$85.95_{0.14}$[b]

The F1 scores of the single-task learning (STL) baselines as shown under the SBD w/ PoS column in the first panel for readability. The bold and underlines denote the best and second best results. The numbers on the subscripts are standard deviations

[a]indicates the models re-implemented by us. w/CL means our CL mechanism is applied

[b]denotes the improvement is statistically significant ($p < 0.01$ on a two-tailed t-test), against the highest baseline score

Table 3 Text chunking results when MTL with PoS tagging (Chunking w/ PoS) and with SBD (Chunking w/ SBD)

Model	Chunking w/ PoS		Chunking w/ SBD	
	F1	Acc	F1	Acc
GRU-CRF	95.41	–	–	–
MVCRF	95.44	–	–	–
Star-C	95.93	–	–	–
Flair-C	96.72	–	–	–
ACE	97.3	–	–	–
Meta	95.11	–	–	–
Gated[a]	97.50	97.63	97.11	97.36
Gated w/CL[a]	<u>97.82</u>	97.94	97.23	97.51
AUX[a]	97.18	97.51	96.69	96.83
AUX w/CL[a]	97.52	97.86	96.80	96.98
Ours-LSTM	$97.96_{0.05}$[b]	$98.04_{0.08}$[b]	$97.40_{0.07}$[b]	$97.73_{0.08}$[b]
Ours-Transformer	$\mathbf{98.13}_{0.06}$[b]	$\mathbf{98.45}_{0.07}$[b]	$97.48_{0.05}$[b]	$97.98_{0.05}$[b]

The F1 scores of the single-task learning baselines as shown under the Chunking w/ PoS column in the first panel for readability. The bold and underlines denote the best and second best results. The numbers on the subscripts are standard deviations

[a]indicates the models re-implemented by us. w/CL means our CL mechanism is applied

[b]denotes the improvement is statistically significant ($p < 0.01$ on a two-tailed t-test), against the highest baseline score

our proposed CL mechanism consistently bring significant improvement to the MTL baselines in all experiment settings, indicating its robustness. Based on the former observation, Our-Transformer is our main model of investigation in the following experiments.

Ablation Study

We conduct an ablation study using the best performing MTL setups for each task, reported in Tables 2, 3, and 4, i.e., chunking paired with PoS tagging, and PoS tagging and SBD paired together. Specially, the following variants are studied:

- **w/o MTL** denotes that the two target tasks are trained on the base tower structure of Transformer encoders and CRF using single-task learning.
- **w/o LGDS** denotes a hard parameter-sharing model without LGDS and CL, where the two tasks share the same encoder layers and keep individual output layers.

Table 4 PoS tagging results when learning with SBD (PoS w/ SBD) and with chunking (PoS w/ chunk)

Model	PoS w/ SBD		PoS w/ chunk	
	F1	Acc	F1	Acc
LSTM-CNN-CRF	–	97.55	–	–
GatedDualCNN	–	97.59	–	–
Star-P	–	97.68	–	–
Flair-P	–	**97.85**	–	–
Meta	–	–	–	97.45
Gated[a]	97.67	97.64	97.47	97.40
Gated w/CL[a]	97.74	97.70	97.60	97.52
AUX[a]	97.61	97.59	97.60	97.53
AUX w/CL[a]	97.71	97.66	97.62	97.56
Ours-LSTM	$\underline{97.78}_{0.02}$[b]	$97.70_{0.03}$	$97.58_{0.03}$	$97.52_{0.03}$
Ours-Transformer	$\mathbf{97.86}_{0.02}$[b]	$\underline{97.79}_{0.02}$	$97.64_{0.04}$	$97.59_{0.03}$

The accuracy of the single-task learning baselines as shown under the PoS w/ SBD column in the first panel for readability. The bold and underlines denote the best and second best results. The numbers on the subscripts are standard deviations

[a]indicates the models re-implemented by us. w/CL means our CL mechanism is applied

[b]denotes the improvement is statistically significant ($p < 0.01$ on a two-tailed t-test), against the highest baseline score

- **Finetune** denotes pre-training the task tower without parallel labels using its corresponding dataset first, then fine-tuning the two task towers with the parallel labeled dataset.
- **w/o CL** denotes that chunking and PoS tagging are trained on the proposed LGDS-based MTL architecture without CL.
- **w/o non-parallel** denotes a w/o CL model that is trained solely on parallel labeled dataset, but is evaluated on testing sets of both tasks.

As seen in Table 5, a hard parameter-sharing MTL model without LGDS (w/o LGDS) yields higher performance than the single task learning model (w/o MTL) on all three tasks. Further comparisons between the performance of w/o MTL and Ours-Transformer in Tables 2, 3, and 4 show that all pair-wise MTL combinations perform better than the single task counterparts. This shows that joint training between syntactic tasks can provide useful features for each other. Learning multiple tasks simultaneously can also help the model against overfitting [68], because the model needs to learn robust representations to achieve the training targets of both tasks. The improvements of w/o CL over w/o LGDS are consistent across the three tasks, showing the effectiveness of our proposed soft parameter-sharing mechanism and layer connections. Comparing the performance of Finetune with w/o CL and Our-Transformer on the three tasks, we can conclude that fine-tuning cannot achieve stable learning for MTL. Next, there is a sharp drop on SBD performance by simply using the WSJ dataset for joint learning with PoS tagging (w/o non-parallel). As a result, the PoS tagging performance also decreases. The same reason can be inferred to be responsible for the performance degradation of Chunking w/ PoS when solely using the CoNLL00 training set for the MTL of text chunking and PoS tagging. This shows the significance of introducing data from both tasks to support the pair-wise MTL. Finally, using the proposed LGDS, training strategies, and CL mechanism can help the model achieve further improvements on the three tasks, which is evidenced by the consistent improvements of Ours-Transformer over w/o CL models across different tasks.

Curriculum Learning Analysis

Figure 4 shows the loss curves of SBD and PoS tagging in pair-wise MTL, given by CL-4 (Fig. 4a) and CL-1 (Fig. 4b), respectively. CL-1 denotes that we randomly sample the equal number of instances from both task datasets in a batch without using any sample selection criterion. CL-4 employs our recommended CL sample size and sample selection criterion. Similarly, Fig. 5 shows the loss curves of PoS tagging and text chunking, and Fig. 6 the curves of SBD and chunking in pair-wise MTL. It can be observed in Fig. 4 that the fluctuation of the PoS tagging loss curve of CL-4 is less than those of CL-1 (the blue lines). The same can be seen in the SBD curves (the green lines), albeit to a smaller extent. It shows that our proposed curriculum criterion (11) is effective in selecting complementary PoS tagging instances to optimize and stabilize the learning of both PoS tagging and SBD. Such smoothing effect can also be observed in Figs. 5 and 6 for the chunking loss curves (the red lines). The difference of PoS tagging loss curves (the blue lines) in the two figures is not conspicuous, as does the difference of SBD loss curves (the green lines). It might be that the learning of chunking does not cause significant biases for that of PoS tagging nor SBD. The comparatively stable loss curves in CL-4 (Figs. 4a, 5a, and 6a) prove our hypothesis (H1 in introduction) that a model can achieve more robust MTL with non-parallel labeled data, if input instances from two different tasks are in similar vector spaces.

We further analyze the tradeoff between time costs and performance using text chunking and PoS tagging as a case study in Table 6. We use the averaged accuracy of chunking and PoS tagging as the overall accuracy measure, because our early stop point is determined by the condition that the highest overall accuracy (the sum of text chunking and PoS tagging accuracy) is not improved in 5 training epochs. We use the CL sample size of 1 (CL-1) as the time baseline, which means we randomly sample a CoNLL00 instance to learn with a WSJ

Table 5 Ablation study results

Model	Chunking w/ PoS		PoS w/ SBD		SBD w/ PoS	
	F1	Acc	F1	Acc	F1	Acc
w/o MTL	95.77	96.64	97.51	97.46	85.04	85.18
w/o LGDS	96.47	97.20	97.61	97.55	86.42	86.57
Finetune	97.21	97.38	97.10	96.84	81.97	82.22
w/o CL	97.71	98.24	97.77	97.68	87.16	87.32
w/o non-parallel	97.35	97.96	96.32	96.33	79.98	80.26
Ours-Transformer	98.13	98.45	97.86	97.79	87.25	87.48

Chunking w/ PoS denotes chunking results when paired with PoS tagging. PoS w/ SBD denotes PoS tagging results when paired with SBD. SBD w/ PoS denotes SBD results when paired with PoS tagging

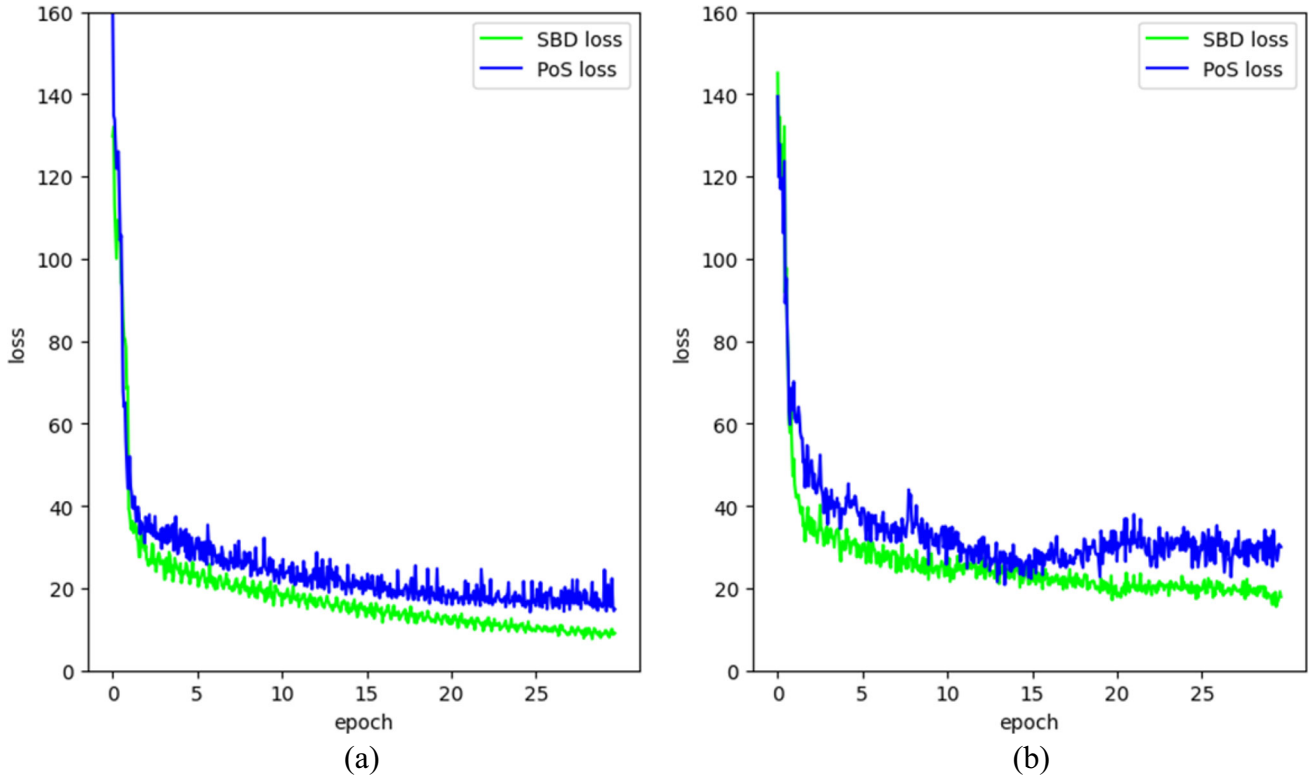

Fig. 4 Loss curves of SBD and PoS tagging, given by **a** our proposed CL mechanism with a sample size in 4; **b** the CL mechanism with a sample size in 1

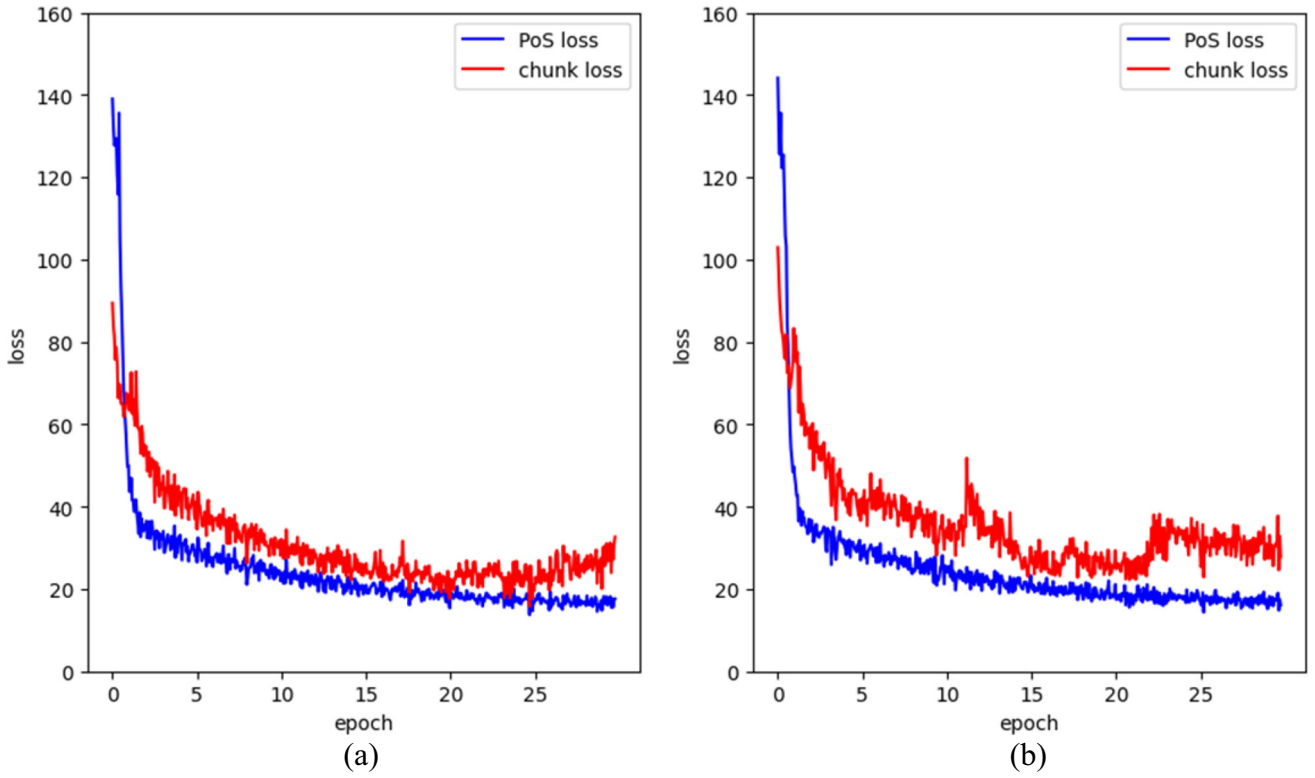

Fig. 5 Loss curves of PoS tagging and text chunking, given by **a** our proposed CL mechanism with a sample size in 4; **b** the CL mechanism with a sample size in 1

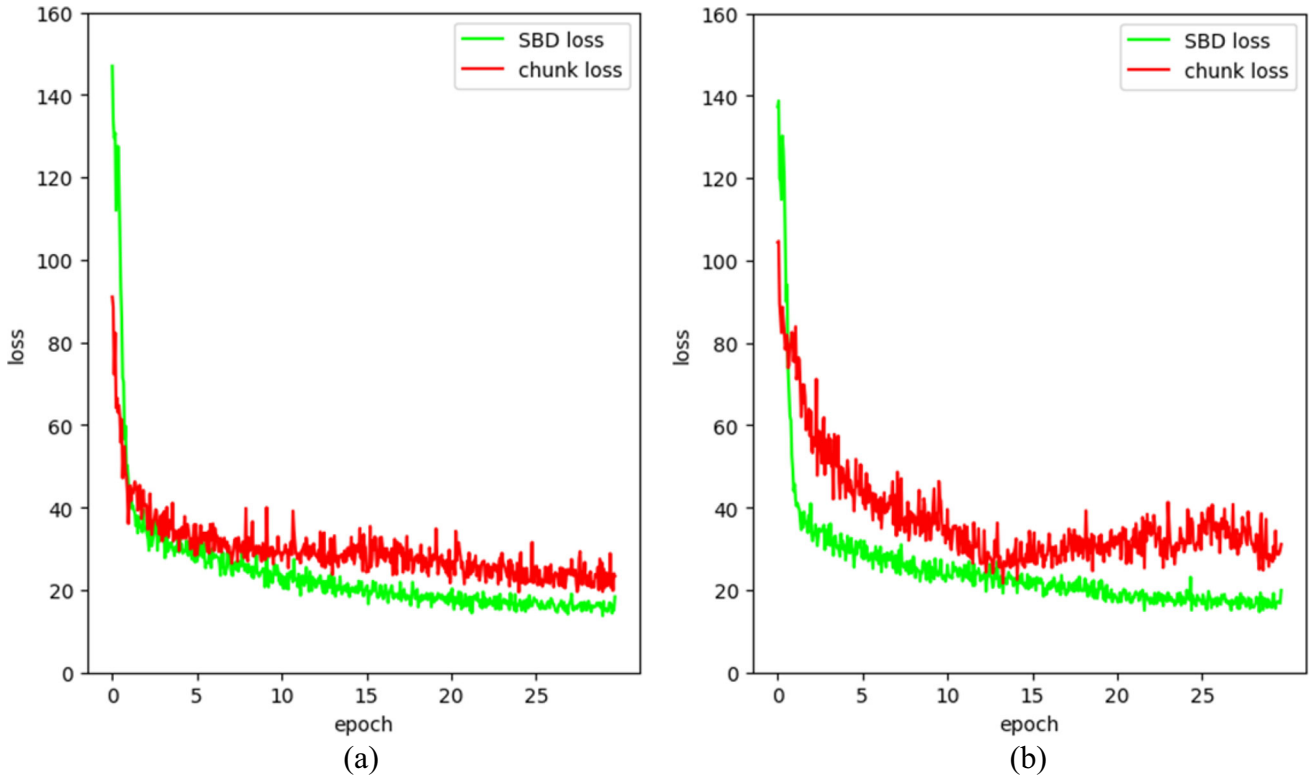

Fig. 6 Loss curves of SBD and text chunking, given by **a** our proposed CL mechanism with a sample size in 4; **b** the CL mechanism with a sample size in 1

instance; CL-2 means the random sample size is 2, and so on. The accuracy shows an upper trend as the sample size grows. However, the time costs also increase, because the model needs to compute and compare more cross-entropy of hidden states when the sample size is larger. The CL improvements in different setups are marginal in our full model, because the model with LGDS and without CL (w/o CL) has achieved very high accuracy in both tasks. Improving model performance is very hard, given the w/o CL baseline has yielded an average accuracy by 97.70%. Compared with the improvement space (2.30%) to the ground-truth (100%), the gap (0.27%) between CL-4 (97.97%) and w/o CL (97.70%) and the gap (0.24%) between CL-4 (97.97%) and CL-1 (97.73%) are reasonable.

Table 6 Curriculum learning sample size analysis by time costs and average accuracy gains. Underlines denote baselines

Setup	Time costs	Δ	Avg acc
w/o CL	0.86X	−0.03	97.70
CL-1	1.00X	–	97.73
CL-2	1.08X	+0.15	97.88
CL-4	1.31X	+0.24	97.97
CL-8	1.74X	+0.26	97.99

Conclusion

In this work, we propose a soft parameter-sharing mechanism to share dependency information that is learned from the two involved tasks in pair-wise MTL. It consists of CNN and Biaffine attention to capturing local and global dependency, respectively. Additionally, we propose a CL mechanism to achieve robust MTL with non-parallel labeled data. The addition of CL mitigates the learning bias given by the task with non-parallel data, so that the performance of both tasks may further improve. The employed curriculum criterion enables effective selection of complementary data, so that the learning loss of the tasks involved can converge more steadily.

Using the proposed MTL method, we conduct a study on how syntactic tasks of different granularity complement each other through pair-wise MTL. We conclude that fine-grained tasks can provide information that yields significant gains for coarse-grained tasks. On the other hand, the benefits that coarse-grained tasks bring to fine-grained tasks are limited, with the exception of SBD to PoS tagging, which is likely because the delineation of structure in an input sequence learned in the SBD task helps the PoS tagging tower focus on learning features specific to sentences, facilitating the learning of long-range label dependencies.

Additionally, our model achieves state-of-the-art performance on text chunking and comparable performance on SBD and PoS tagging to the state-of-the-art baselines. We will test if our CL mechanism can relax task relevance requirements in MTL in future work.

Author Contributions X.L. and R.M. both contributed to the conceptualization and methodology of the study. X.L. conducted the experiments, and wrote the manuscript. R.M. revised the manuscript. E.C. supervised the study, and edited and approved the final version of the manuscript.

Data Availability No datasets were generated or analysed during the current study.

Declarations

Ethics Approval This paper does not contain any studies with human participants or animals performed by any of the authors.

Competing Interests The authors declare no competing interests.

References

1. Woolf BP. Chapter 5 - Communication knowledge. In: Woolf BP, editor. Building intelligent interactive tutors. San Francisco: Morgan Kaufmann; 2009. pp. 136–82.
2. Cambria E, Mao R, Chen M, Wang Z, Ho S-B. Seven pillars for the future of Artificial Intelligence. IEEE Intell Syst. 2023;38(6):62–9.
3. Matsoukas S, Bulyko I, Xiang B, Nguyen K, Schwartz R, Makhoul J. Integrating speech recognition and machine translation. In: 2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07 (vol. 4). IEEE; 2007. p. 1281.
4. Zhou N, Wang X, Aw A. Dynamic boundary detection for speech translation. In: Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC) (vol. 2017). IEEE; 2017. pp. 651–6.
5. Krallinger M, Rabal O, Lourenco A, Oyarzabal J, Valencia A. Information retrieval and text mining technologies for chemistry. Chemical Rev. 2017;117(12):7673–761.
6. Jing H, Lopresti D, Shih C. Summarization of noisy documents: a pilot study. In: Proceedings of the HLT-NAACL 03 Text Summarization Workshop. 2003. pp. 25–32.
7. Boudin F, Huet S, Torres-Moreno J-M. A graph-based approach to cross-language multi-document summarization. Polibits. 2011;43:113–8.
8. Councill I, McDonald R, Velikovich L. What's great and what's not: Learning to classify the scope of negation for improved sentiment analysis. In: Proceedings of the Workshop on Negation and Speculation in Natural Language Processing. 2010. pp. 51–9.
9. Gupta H, Kottwani A, Gogia S, Chaudhari S. Text analysis and information retrieval of text data. In: 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET). IEEE; 2016. pp. 788–92.
10. Syed AZ, Aslam M, Martinez-Enriquez AM. Associating targets with SentiUnits: a step forward in sentiment analysis of Urdu text. Artif Intell Rev. 2014;41(4):535–61.
11. Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P. Natural language processing (almost) from scratch. J Mach Learn Res. 2011;12:2493–537.
12. Sun X, Sun S, Yin M, Yang H. Hybrid neural conditional random fields for multi-view sequence labeling. Knowl-Based Syst. 2020;189:105151.
13. Dozat T, Manning CD. Deep biaffine attention for neural dependency parsing. arXiv:1611.01734 [Preprint]. 2016. Available from: http://arxiv.org/abs/1611.01734.
14. Zhou H, Zhang Y, Li Z, Zhang M. Is POS tagging necessary or even helpful for neural dependency parsing? 2020.
15. Mahmood A, Khan HU, Zahoor-ur-Rehman, Khan W. Query based information retrieval and knowledge extraction using hadith datasets. In: 2017 13th International Conference on Emerging Technologies (ICET). 2017. pp. 1–6. https://doi.org/10.1109/ICET.2017.8281714.
16. Asghar MZ, Khan A, Ahmad S, Kundi FM. A review of feature extraction in sentiment analysis. J Basic Appl Scientific Res. 2014;4(3):181–6.
17. Cambria E, Zhang X, Mao R, Chen M, Kwok K. SenticNet 8: Fusing emotion AI and commonsense AI for interpretable, trustworthy, and explainable affective computing. In: Proceedings of the 26th International Conference on Human-computer Interaction (HCII). 2024.
18. Mao R, Lin C, Guerin F. Word embedding and WordNet based metaphor identification and interpretation. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (vol. 1). 2018. pp. 1222–31.
19. Ge M, Mao R, Cambria E. Explainable metaphor identification inspired by conceptual metaphor theory. In: Proceedings of AAAI. 2022. pp. 10681–9.
20. Mao R, Li X, He K, Ge M, Cambria E. MetaPro Online: a computational metaphor processing online system. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations). Toronto: Association for Computational Linguistics; 2023. pp. 127–35. https://aclanthology.org/2023.acl-demo.12.
21. Akbik A, Blythe D, Vollgraf R. Contextual string embeddings for sequence labeling. In: Proceedings of the 27th International Conference on Computational Linguistics. 2018. pp. 1638–49.
22. Wang X, Jiang Y, Bach N, Wang T, Huang Z, Huang F, Tu K. Automated concatenation of embeddings for structured prediction. arXiv:2010.05006 [Preprint]. 2020. Available from: http://arxiv.org/abs/2010.05006.
23. Wong DF, Chao LS, Zeng X. iSentenizer-: Multilingual sentence boundary detection model. Scientific World J. 2014;2014.
24. Zhang X, Mao R, Cambria E. A survey on syntactic processing techniques. Artif Intell Rev. 2023;56(6):5645–728.
25. Chen J, Qiu X, Liu P, Huang X. Meta multi-task learning for sequence modeling. In: Proceedings of the AAAI Conference on Artificial Intelligence. 2018. p. 32.
26. Yang Z, Salakhutdinov R, Cohen WW. Transfer learning for sequence tagging with hierarchical recurrent networks. arXiv:1703.06345 [Preprint]. 2017. Available from: http://arxiv.org/abs/1703.06345.
27. Bender E.M, Koller A. Climbing towards NLU: On meaning, form, and understanding in the age of data. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020. pp. 5185–98.
28. Mao R, Chen G, Zhang X, Guerin F, Cambria E. GPTEval: A survey on assessments of ChatGPT and GPT-4. In: Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING), Torino, Italia. 2024. pp. 7844–66.
29. Cambria E, Poria S, Gelbukh A, Thelwall M. Sentiment analysis is a big suitcase. IEEE Intell Syst. 2017;32(6):74–80. https://doi.org/10.1109/MIS.2017.4531228.

30. Marcus MP, Santorini B, Marcinkiewicz MA. Building a large annotated corpus of English: the Penn Treebank. Comput Linguist. 1993;19(2):313–30.
31. Ratinov L, Roth D. Design challenges and misconceptions in named entity recognition. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009). 2009. pp. 147–55.
32. Che X, Wang C, Yang H, Meinel C. Punctuation prediction for unsegmented transcript based on word vector. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16). 2016. pp. 654–58.
33. Mao R, Li X. Bridging towers of multi-task learning with a gating mechanism for aspect-based sentiment analysis and sequential metaphor identification. Proc AAAI Conf Artif Intell. 2021;35:13534–42.
34. Ruder S. An overview of multi-task learning in deep neural networks. arXiv:1706.05098 [Preprint]. 2017. Available from: http://arxiv.org/abs/1706.05098.
35. Chen S, Zhang Y, Yang Q. Multi-task learning in natural language processing: an overview. arXiv:2109.09138 [Preprint]. 2021. Available from: http://arxiv.org/abs/2109.09138.
36. Sang EF, Buchholz S. Introduction to the CoNLL-2000 shared task: chunking. In: Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop. ConLL '00. Association for Computational Linguistics; 2000. pp. 127–32. https://doi.org/10.3115/1117601.1117631.
37. Le D, Thai M, Nguyen T. Multi-task learning for metaphor detection with graph convolutional neural networks and word sense disambiguation. In: Proceedings of the 34th AAAI Conference on Artificial Intelligence. 2020. pp. 8139–46.
38. Zhang Z, Yu W, Yu M, Guo Z, Jiang M. A survey of multi-task learning in natural language processing: regarding task relatedness and training methods. arXiv:2204.03508 [Preprint]. 2022. Available from: http://arxiv.org/abs/2204.03508.
39. Bhat S, Debnath A, Banerjee S, Shrivastava M. Word embeddings as tuples of feature probabilities. In: Proceedings of the 5th Workshop on Representation Learning for NLP. Association for Computational Linguistics, Online; 2020. pp. 24–33. https://doi.org/10.18653/v1/2020.repl4nlp-1.4, https://aclanthology.org/2020.repl4nlp-1.4.
40. Grefenstette G, Tapanainen P. What is a word, what is a sentence? Problems of tokenisation. Report, Grenoble Laboratory; 1994.
41. Stamatatos E, Fakotakis N, Kokkinakis G. Automatic extraction of rules for sentence boundary disambiguation. In: Proceedings of the Workshop on Machine Learning in Human Language Technology. Citeseer; 1999. pp. 88–92.
42. Sadvilkar N, Neumann M. PySBD: pragmatic sentence boundary disambiguation. arXiv:2010.09657 [Preprint]. 2020. Available from: http://arxiv.org/abs/2010.09657.
43. Knoll BC, Lindemann EA, Albert AL, Melton GB, Pakhomov SVS. Recurrent deep network models for clinical NLP tasks: Use case with sentence boundary disambiguation. Stud Health Technol Inf. 2019;264(31437913):198–202. https://doi.org/10.3233/SHTI190211.
44. Makhija K, Ho T-N, Chng E-S. Transfer learning for punctuation prediction. In: Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC) (vol. 2019). IEEE; 2019. pp. 268–73.
45. Alam T, Khan A, Alam F. Punctuation restoration using transformer models for high-and low-resource languages. In: Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020). 2020. pp. 132–42.
46. Palmer DD, Hearst MA. Adaptive multilingual sentence boundary disambiguation. Comput Linguist. 1997;23(2):241–67.
47. Mikheev A. Tagging sentence boundaries. In: 1st Meeting of the North American Chapter of the Association for Computational Linguistics. 2000.
48. Agarwal N, Ford KH, Shneider M. Sentence boundary detection using a maxEnt classifier. In: Proceedings of MISC. 2005. pp. 1–6.
49. Ramshaw LA, Marcus M. Text chunking using transformation-based learning. In: Yarowsky D, Church K, editors. Third Workshop on Very Large Corpora. 1995. https://aclanthology.org/W95-0107/.
50. Sutton C, McCallum A, Rohanimanesh K. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. J Mach Learn Res. 2007;8(3).
51. Sun X, Morency L-P, Okanohara D, Tsuruoka Y, Tsujii J. Modeling latent-dynamic in shallow parsing: a latent conditional model with improved inference. In: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008). 2008. pp. 841–8.
52. Lin JC-W, Shao Y, Zhang J, Yun U. Enhanced sequence labeling based on latent variable conditional random fields. Neurocomputing. 2020;403:431–40.
53. Liu Y, Li G, Zhang X. Semi-Markov CRF model based on stacked neural Bi-LSTM for sequence labeling. In: 2020 IEEE 3rd International Conference of Safe Production and Informatization (IICSPI). 2020. pp. 19–23. https://doi.org/10.1109/IICSPI51290.2020.9332321.
54. Huang Z, Xu W, Yu K. Bidirectional LSTM-CRF models for sequence tagging. arXiv:1508.01991 [Preprint]. 2015. Available from: http://arxiv.org/abs/1508.01991.
55. Rei M. Semi-supervised multitask learning for sequence labeling. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vancouver: Association for Computational Linguistics; 2017. pp. 2121–30. https://doi.org/10.18653/v1/P17-1194, https://aclanthology.org/P17-1194.
56. Zhai F, Potdar S, Xiang B, Zhou B. Neural models for sequence chunking. arXiv:1701.04027 [Preprint]. 2017. Available from: http://arxiv.org/abs/1701.04027.
57. Yang Z, Salakhutdinov R, Cohen W. Multi-task cross-lingual sequence tagging from scratch. arXiv:1603.06270 [Preprint]. 2016. Available from: http://arxiv.org/abs/1603.06270.
58. Wei W, Wang Z, Mao X, Zhou G, Zhou P, Jiang S. Position-aware self-attention based neural sequence labeling. Pattern Recognit. 2021;110:107636.
59. Church KW. A stochastic parts program and noun phrase parser for unrestricted text. In: Second Conference on Applied Natural Language Processing. Austin: Association for Computational Linguistics; 1988. pp. 136–43. https://doi.org/10.3115/974235.974260, https://www.aclweb.org/anthology/A88-1019.
60. Kupiec J. Robust part-of-speech tagging using a hidden Markov model. Comput Speech Lang. 1992;6(3):225–42. https://doi.org/10.1016/0885-2308(92)90019-Z.
61. Brants T. TnT-a statistical part-of-speech tagger. arXiv:cs/0003055 [Preprint]. 2000. Available from: http://arxiv.org/abs/cs/0003055.
62. McCallum A, Freitag D, Pereira FC. Maximum entropy Markov models for information extraction and segmentation. In: Icml (vol. 17). 2000. pp. 591–8.
63. Lafferty JD, McCallum A, Pereira FCN. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning. ICML '01. San Francisco: Morgan Kaufmann Publishers Inc.; 2001. pp. 282–9.
64. Dos Santos C, Zadrozny B. Learning character-level representations for part-of-speech tagging. In: International Conference on Machine Learning. PMLR; 2014. pp. 1818–26.

65. Ma X, Hovy E. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. arXiv:1603.01354 [Preprint]. 2016. Available from: http://arxiv.org/abs/1603.01354.
66. Chiu JP, Nichols E. Named entity recognition with bidirectional LSTM-CNNs. Trans Assoc Computat Linguist. 2016;4:357–70.
67. Zhao L, Qiu X, Zhang Q, Huang X. Sequence labeling with deep gated dual path CNN. IEEE/ACM Trans Audio Speech Lang Process. 2019;27(12):2326–35.
68. Ruder S. An overview of multi-task learning in deep neural networks. arXiv:1706.05098 [Preprint]. 2017. Available from: http://arxiv.org/abs/1706.05098.
69. Ma Y, Mao R, Lin Q, Wu P, Cambria E. Quantitative stock portfolio optimization by multi-task learning risk and return. Inf Fusion. 2024;104:102165. https://doi.org/10.1016/j.inffus.2023.102165.
70. He K, Mao R, Gong T, Li C, Cambria E. Meta-based self-training and re-weighting for aspect-based sentiment analysis. IEEE Trans Affective Comput. 2023;14(3):1731–42. https://doi.org/10.1109/TAFFC.2022.3202831.
71. Liu P, Qiu X, Huang X. Recurrent neural network for text classification with multi-task learning. arXiv:1605.05101 [Preprint]. 2016. Available from: http://arxiv.org/abs/1605.05101.
72. Zhao S, Liu T, Zhao S, Wang F. A neural multi-task learning framework to jointly model medical named entity recognition and normalization. In: Proceedings of the AAAI Conference on Artificial Intelligence (vol. 33). 2019. pp. 817–24.
73. Soviany P, Ionescu RT, Rota P. Sebe N. Curriculum learning: a survey. Int J Comput Vis. 2022:1–40.
74. Ma F, Meng D, Xie Q, Li Z, Dong X. Self-paced co-training. In: International Conference on Machine Learning. PMLR; 2017. pp. 2275–84.
75. Zhang X, Kumar G, Khayrallah H, Murray K, Gwinnup J, Martindale MJ, McNamee P, Duh K, Carpuat M. An empirical exploration of curriculum learning for neural machine translation. arXiv:1811.00739 [Preprint]. 2018. Available from: http://arxiv.org/abs/1811.00739.
76. Wang W, Caswell I, Chelba C. Dynamically composing domain-data selection with clean-data selection by "co-curricular learning" for neural machine translation. arXiv:1906.01130 [Preprint]. 2019. Available from: http://arxiv.org/abs/1906.01130.
77. Kocmi T, Bojar O. Curriculum learning and minibatch bucketing in neural machine translation. In: Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP (vol. 2017). 2017. pp. 379–86.
78. Liu C, He S, Liu K, Zhao J, et al. Curriculum learning for natural answer generation. In: IJCAI. 2018. pp. 4223–9.
79. Wu L, Tian F, Xia Y, Fan Y, Qin T, Jian-Huang L, Liu T-Y. Learning to teach with dynamic loss functions. Adv Neural Inf Process Syst. 2018;31.
80. Hacohen G, Weinshall D. On the power of curriculum learning in training deep networks. In: International Conference on Machine Learning. PMLR; 2019. pp. 2535–44.
81. Zhang M, Yu Z, Wang H, Qin H, Zhao W, Liu Y. Automatic digital modulation classification based on curriculum learning. Appl Sci. 2019;9(10):2171.
82. Sangineto E, Nabi M, Culibrk D, Sebe N. Self paced deep learning for weakly supervised object detection. IEEE Trans Pattern Anal Mach Intell. 2018;41(3):712–25.
83. Kim D, Bae J, Jo Y, Choi J. Incremental learning with maximum entropy regularization: rethinking forgetting and intransigence. arXiv:1902.00829 [Preprint]. 2019. Available from: http://arxiv.org/abs/1902.00829.
84. Castells T, Weinzaepfel P, Revaud J. Superloss: a generic loss for robust curriculum learning. Adv Neural Inf Process Syst. 2020;33:4308–19.
85. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. Adv Neural Inf Process Syst. 2017;30.
86. Khan S, Naseer M, Hayat M, Zamir SW, Khan FS, Shah M. Transformers in vision: a survey. ACM Comput Surv (CSUR). 2021.
87. Mao R, Li X, Ge M, Cambria E. Metapro: a computational metaphor processing model for text pre-processing. Inf Fusion. 2022;86–87:30–43. https://doi.org/10.1016/j.inffus.2022.06.002.
88. Forney GD. The Viterbi algorithm. Proc IEEE. 1973;61(3):268–78.
89. Tilk O, Alumäe T. Bidirectional recurrent neural network with attention mechanism for punctuation restoration. In: Interspeech (vol. 3). 2016. p. 9.
90. Guo Q, Qiu X, Liu P, Shao Y, Xue X, Zhang Z. Star-transformer. arXiv:1902.09113 [Preprint]. 2019. Available from: http://arxiv.org/abs/1902.09113.
91. Pennington J, Socher R, Manning CD. Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014. pp. 1532–43.
92. Devlin J, Chang M-W, Lee K, Toutanova K. Bert: pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805 [Preprint]. 2018. Availalble from: http://arxiv.org/abs/1810.04805.
93. Dankers V, Rei M, Lewis M, Shutova E. Modelling the interplay of metaphor and emotion through multitask learning. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2019. pp. 2218–29.
94. Alqahtani S, Mishra A, Diab M. A multitask learning approach for diacritic restoration. arXiv:2006.04016 [Preprint]. 2020. Available from: http://arxiv.org/abs/2006.04016.
95. Collins M. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In: Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing, EMNLP 2002, Philadelphia, PA, USA, July 6-7, 2002. pp. 1–8. https://doi.org/10.3115/1118693.1118694, https://aclanthology.org/W02-1001/.
96. Kingma DP, Ba J. Adam: a method for stochastic optimization. arXiv:1412.6980 [Preprint]. 2014. Available from: http://arxiv.org/abs/1412.6980.