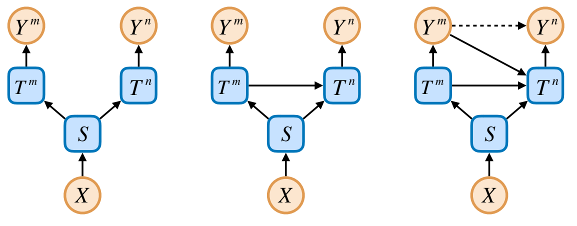
Exploring Logically Dependent Multi-task Learning with Causal Inference

Wenqing Chen, Jidong Tian, Liqiang Xiao, Hao He*, Yaohui Jin*
MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
State Key Lab of Advanced Optical Communication System and Network,
Shanghai Jiao Tong University
`{wenqingchen, hehao, jinyh}@sjtu.edu.cn`

Abstract

Previous studies have shown that hierarchical multi-task learning (MTL) can utilize task dependencies by stacking encoders and outperform democratic MTL. However, stacking encoders only considers the dependencies of feature representations and ignores the label dependencies in logically dependent tasks. Furthermore, how to properly utilize the labels remains an issue due to the cascading errors between tasks. In this paper, we view logically dependent MTL from the perspective of causal inference and suggest a mediation assumption instead of the confounding assumption in conventional MTL models. We propose a model including two key mechanisms: label transfer (LT) for each task to utilize the labels of all its lower-level tasks, and Gumbel sampling (GS) to deal with cascading errors. In the field of causal inference, GS in our model is essentially a counterfactual reasoning process, trying to estimate the causal effect between tasks and utilize it to improve MTL. We conduct experiments on two English datasets and one Chinese dataset. Experiment results show that our model achieves state-of-the-art on six out of seven subtasks and improves predictions' consistency.



(a) Democratic (b) Hierarchical (c) Logically Dependent

Figure 1: Three types of MTL schemes for two tasks t_m and t_n: (a) Democratic MTL shares an encoder S and owns task-specific encoders T^i ($i = m$ or n); (b) Hierarchical MTL stacks those task-specific encoders; (c) Logically dependent MTL further considers the label dependencies by re-utilizing low-level tasks' labels.

1 Introduction

Multi-task learning (MTL) has received increasing interest with the knowledge transfer potential among related tasks (Caruana, 1997; Ruder et al., 2017; Liu et al., 2019). Recently, hierarchical MTL models (Hashimoto et al., 2017; Sanh et al., 2018) were proposed for tasks with dependencies and achieved better performance than democratic ones. In their models, the encoders of different tasks were stacked. And the proposition is that complex tasks at top layers require deep processes to capture semantic richer features, and simple tasks at bottom layers require shallow processes.

However, many hierarchical MTL models only consider the dependencies of feature representations and ignore the label dependencies. A direct comparison is shown in Figure 1. Let X denote a given input, and Y^m and Y^n denote the outputs of two tasks, respectively. The first two MTL schemes (Liu et al., 2017; Zheng et al., 2018; Sanh et al., 2018) substantially model the likelihood by $P(Y^m, Y^n|X)=P(Y^m|X)P(Y^n|X)$ while the third MTL scheme (Bekoulis et al., 2018; Luan et al., 2019) models it by: $P(Y^m, Y^n|X)=P(Y^m|X)P(Y^n|X, Y^m)$. From the causal perspective, the first two schemes assume that Y^m and Y^n are conditional independent, while the third scheme assumes that Y^m has a causal effect on Y^n. In this paper, we suggest that the causal effect is important for logically dependent tasks. And we propose a mechanism referred to label transfer (LT), which lets a task utilize the labels of all its lower-level tasks.

When utilizing discrete labels, there remains another issue. For example, the strategy in (Bek-

* Corresponding authors

oulis et al., 2018) used gold labels of low-level tasks during training and used predicted ones during testing. Apparently, there was a train-test discrepancy which leads to cascading errors between tasks. And it was similar to the exposure bias problem in the field of text generation (Ranzato et al., 2016). Recently, two approaches have been investigated to overcome the problem, which are Gumbel Sampling (GS) (Kusner and Hernández-Lobato, 2016; Nie et al., 2019) and Reinforcement Learning (RL) (Yu et al., 2017; Guo et al., 2018). In this paper, we regard the logically dependent MTL as a task-level label generation problem. And we incorporate GS because it feeds the optimizer with low-variance gradients, improving stability and speed of training over RL. Specifically, our model samples a label from the predicted probability distribution for each task and feeds it to its higher-level tasks. And back-propagated gradients will penalize wrong predictions if the causal effect exists. From the perspective of causal inference, the sampling is a counterfactual reasoning process that can estimate the causal effect between different tasks' labels. And we hope a model properly cooperating causality will be more robust and transferable, as argued in (Schölkopf, 2019).

To verify the effectiveness of our model, we conduct experiments on two English and one Chinese MTL datasets. The results show that our model achieves state-of-the-art (SOTA) on 6 out of 7 subtasks and improves predictions' consistency. And we present the estimated causal effect for several cases, which is consistent with humans' prior knowledge. In conclusion, the contributions of this paper can be summarized as follows:

- We view MTL from the causal perspective and suggest a mediation assumption instead of the confounding assumption in conventional MTL models.

- We propose a novel MTL model with two key mechanisms: label transfer and Gumbel sampling, which better utilize task dependencies and alleviate cascading errors.

- The experiments are carried on both English and Chinese datasets and demonstrate our model's effectiveness and better consistency of predicted results for subtasks.

2 Related Work

In natural language processing (NLP), many studies focus on modeling task dependencies to improve MTL. A line of work proposed hierarchical MTL architectures by stacking the encoders of different tasks with simple tasks at lower layers and complex tasks at top layers (Sgaard and Goldberg, 2016; Hashimoto et al., 2017; Sanh et al., 2018). And Zhong et al. (2018) proposed a topological MTL architecture based on the topological hierarchy of tasks. Another line of work tried to re-encode the predictions of low-level tasks. Giannis et al. (2018) re-encoded the predicted labels with the highest probability of low-level tasks during testing, and encoded the gold labels during training. Luan et al. (2019) re-encoded the soft predictions during testing and also encoded the gold ones during training. Yang et al. (2019) proposed a bidirectional architecture producing initial probability distributions for different tasks and then refine the probability distributions by conditioning on each other during both training and testing.

Our work is also related to some studies for text generation. The democratic and hierarchical MTL schemes in Figure 1 are similar to the non-autoregressive language models like BERT (Devlin et al., 2019) which is possible to generate syntactically incorrect sentences (Ghazvininejad et al., 2019). The logically dependent MTL scheme is similar to the autoregressive language model but remains a train-test discrepancy. GS or RL learning have been investigated to deal with the discrepancy (Kusner and Hernández-Lobato, 2016; Yu et al., 2017; Guo et al., 2018; Nie et al., 2019).

3 Task Definition

To show that our method's high versatility, we investigate three different MTL scenarios: joint entity and relation extraction (JERE), aspect-based sentiment analysis (ABSA), and legal judgment prediction (LJP).

3.1 Joint Entity and Relation Extraction

JERE includes entity mention extraction (EMD) and relation extraction (RE) (Li and Ji, 2014).

Entity Mention Extraction. EMD can be formulated as a sequence labeling problem with a BILOU scheme (Sanh et al., 2018). Given a sequence of tokens $X = \{x_1, x_2, ..., x_n\}$, EMD assigns a categorical label to each token $Y^{(e)} =$

$\left\{y_1^{(e)}, y_2^{(e)}, ..., y_n^{(e)}\right\}, y_i^{(e)} \in \mathcal{C}_{(e)}$ where $\mathcal{C}_{(e)}$ denotes a predefined set of categories.

Relation Extraction. RE can be formulated as a multi-head selection problem (Bekoulis et al., 2018). For the given sentence X, RE will output a three-dimensional matrix $Y^{(r)} = \{y_{i,j,c}^{(r)}\}, i \leq n, j \leq n, c \in \mathcal{C}_{(r)}$ where $y_{i,j,k}^{(r)}$ denotes a binary value representing the existence of the cth relation between the ith and the jth tokens, and $\mathcal{C}_{(r)}$ denotes the set of categories. Consistent with (Bekoulis et al., 2018), we only consider relations between the last token of the head entity mentions. Redundant relations are therefore not classified.

3.2 Aspect Based Sentiment Analysis

The challenge of "SemEval-2014 Task 4" divides ABSA into four subtasks (Pontiki et al., 2014), and we consider two subtasks which are logically dependent.

Aspect Term Extraction. ATE can also be formulated as a sequence labeling problem with BILOU scheme (Li et al., 2019) or BIO scheme (Luo et al., 2019b; He et al., 2019). Similar to EMD, ATE assigns a categorical label to each token $Y^{(t)} = \left[y_1^{(t)}, y_2^{(t)}, ..., y_n^{(t)}\right], y_i^{(t)} \in \mathcal{C}_{(t)}$ where $\mathcal{C}_{(t)}$ denotes a predefined set of categories.

Aspect Category Detection. ACD is to detect the aspect categories for the given sentence X, which can be formulated as a multi-label classification problem. Let $\mathcal{C}_{(d)}$ denotes a predefined set of categories, we need to predict a label set $Y^{(d)} = \{y_c^{(d)}\}, c \in \mathcal{C}_{(d)}$ for the given sentence X where $y_c^{(d)}$ is a binary value representing existence of the cth category.

3.3 Legal Judgment Prediction

LJP aims to predict the judgment results of legal cases, such as relevant law articles and charges. In this paper, we consider three subtasks for Chinese LJP: **Relevant Article Prediction** (RAP), **Charge Prediction** (CP), and **Prison Term Prediction** (PTP). Following previous work (Zhong et al., 2018; Yang et al., 2019), we only consider those cases with single relevant article and single charge, and divide the prison term into non-overlapping intervals. Then each subtask can be formulated as a single-label classification problem. Specifically, for the given case X, LJP is to assign labels $y^{(a)} \in \mathcal{C}_{(a)}, y^{(c)} \in \mathcal{C}_{(c)}, y^{(p)} \in \mathcal{C}_{(p)}$ where $\mathcal{C}_{(a)}, \mathcal{C}_{(c)}$ and $\mathcal{C}_{(p)}$ are the sets of categories of RAP, CP, and PTP, respectively.

For the three MTL scenarios, the common point is that the subtasks in each are logically dependent. And we have prior knowledge of that the logical orders are EMD→RE, ATE→ACD, and RAP→CP→PTP respectively. And the first scenario is to investigate the knowledge transfer between two token-level tasks, and the second scenario is from a token-level to a sentence-level task. The third is among three sentence-level tasks.

4 Methodology

In this section, we first analyze MTL from the causal perspective in Subsection 4.1 and then introduce our models in the following subsections.

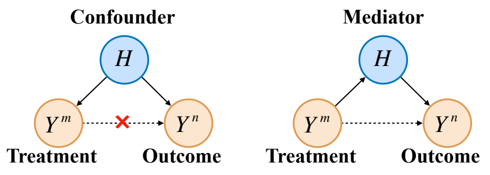


Figure 2: Two kinds of typical causal assumptions. Let H be the feature representation for X, and Y^m and Y^n be the outputs for two tasks t_m and t_n, respectively. The confounding assumption (the left sub-figure) considers Y^m and Y^n to be conditionally independent, while the mediation assumption (the right sub-figure) considers the logical dependency from Y^m to Y^n.

4.1 Basic Causal Assumptions

Let X, Y be two variables representing a sequence of text and the corresponding label, and H be the feature representations of X. The causal graph is therefore $X \rightarrow H \rightarrow Y$. Previous work suggests that MTL may help extract common useful features (Liu et al., 2017), which mainly enhance the process $X \rightarrow H$. When considering $H \rightarrow Y$, there are two possible causal assumptions for MTL: the confounding and mediation shown in Figure 2 where Y^m and Y^n are the outputs of task t_m and t_n respectively.

The confounding assumption is that Y^m and Y^n are conditionally independent and only determined by H. However, For logically dependent tasks, we suggest a mediation assumption that Y^m has a causal effect on Y^n. Specifically, the assumption includes two causal paths between Y^m and Y^n. One links Y^m to Y^n through the mediator H (the solid line), known as the indirect effect. The other links Y^m to Y^n directly (the dashed line), known as the direct effect.

It is worth noting that someone may argue that

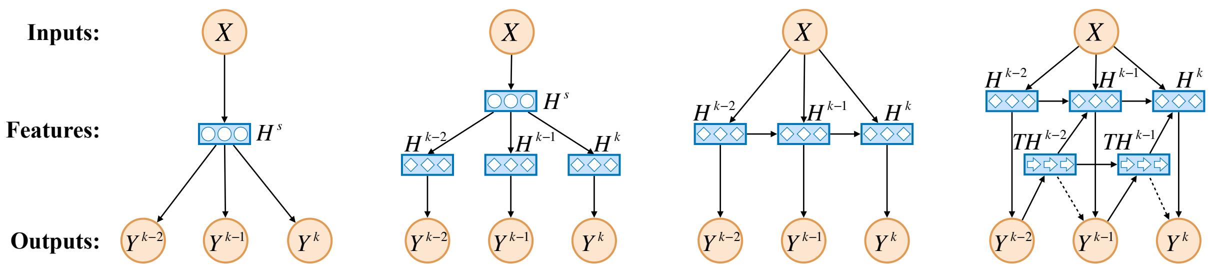


Figure 3: The full causal graphs of four types of MTL models: (a) Fully-Shared MTL shares the feature representation H^s for all tasks; (b) Shared-Private MTL learns a task-specific representation H^k based on H^s for task t_k; (c) Hierarchical MTL stacks the encoders of different tasks; (d) Our model Causal MTL further incorporates the inter-task causality through the indirect causal path, $TH^{k-1}{\rightarrow}H^k{\rightarrow}Y^k$, and the direct causal path, $TH^{k-1}{\rightarrow}Y^k$.

Y^m and Y^n can have mutual causal effects on each other, but the causal graphs are acyclic in most cases. Moreover, recent work has demonstrated that the hierarchical order matters (Sanh et al., 2018).

4.2 Full Causal Graphs

We denote our model as causal multi-task learning (CMTL) and show the full causal graphs in Figure 3 when considering more than two tasks. We also compare with other three typical MTL models, including fully-shared multi-task learning (FSMTL), shared-private multi-task learning (SPMTL), hierarchical multi-task learning (HMTL). FSMTL shares the feature representation H^s for all tasks, and SPMTL learns a task-specific representation H^k based on H^s for task t_k. HMTL stacks the encoders of different tasks. Our model CMTL is derived from HMTL, but the main difference is that CMTL incorporates the inter-task causality through two paths. It first creates an intermediate variable TH^{k-1} conveying the label information of all tasks preceding t_k. Then the model involves the indirect causal effect by the path $TH^{k-1}{\rightarrow}H^k{\rightarrow}Y^k$. And it also involves the direct causal effect by the path $TH^{k-1}{\rightarrow}Y^k$.

4.3 Model Details

The architecture of our model is shown in Figure 4. Generally, the indirect causal effect is implemented by the solid lines connecting "Label Transfer" and "Encoders". And the direct causal effect is implemented by the dashed lines connecting "Label Transfer" and "Predictors".

Token Embedding. Firstly, given an input sentence, X with length n, a token embedding layer is used to map each token into a fixed-dimensional vector. When combining with BERT (Devlin et al.,

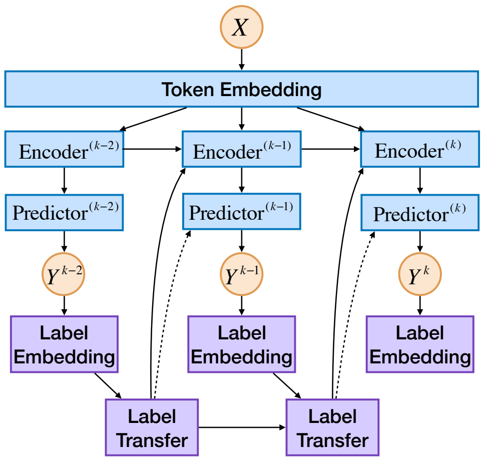


Figure 4: The architecture of CMTL. The indirect and direct causal paths are implemented by the solid lines connecting "Label Transfer" and "Encoders", and the dashed lines connecting "Label Transfer" and "Predictors", respectively.

2019), we will keep it fixed during training to save the cost of memory. And BERT will convert X to a sequence of WordPiece tokens with a length greater than n. To make it suitable for token-level tasks, we select the first WordPiece token embedding for each original token. Furthermore, we use the normalization of different layers, which is similar to ELMo (Peters et al., 2018) to utilize deep and shallow embeddings. The final token embeddings are denoted by $\boldsymbol{E} = \{\boldsymbol{e}_i\}_{1 \le i \le n}$.

Label Embedding. As shown in Figure 4, label embedding layers are used to encode the labels of tasks. We denote the gold label of task t_k as $Y^k = \{y_i^k\}, y_i^k \in \mathcal{C}_k$ where $\mathcal{C}_k$ is the set of categories. Let $1{\le}i{\le}n$ when t_k is a token-level task, and let $i{=}0$ when t_k is a sentence-level task (Y^k contains only one element). Then our model encode Y^k to label embeddings $\boldsymbol{LE}^k = \{\boldsymbol{le}_i^k\}$. Specifically, our model convert y_i^k to a one-hot vector $\boldsymbol{y}_i^k$, and compute the label embedding by:

$$\boldsymbol{le}_i^k = \boldsymbol{W}^k \boldsymbol{y}_i^k \tag{1}$$

where $\boldsymbol{W}^k$ is the parameter matrix for task t_k with shape $d_l \times |\mathcal{C}_k|$ and d_l is the dimension of label embeddings. In this way, the labels of different tasks are mapped into the same latent space.

Label Transfer. After label embedding, for a task, we want our model to utilize the label information of all its preceding tasks instead of only the last one. This process is naturally suitable for recurrent neural networks (RNNs) in which the k-th element depends on its preceding $k-1$ elements. In our model, RNN-LSTM (Hochreiter and Schmidhuber, 1997) is adopted for LT which maintains transferred hidden states $\boldsymbol{TH}^k = \{\boldsymbol{th}_i^k\}$ for task t_k. And the computation of $\boldsymbol{th}_i^k$ can be expressed as:

$$\boldsymbol{th}_i^k = \overrightarrow{LSTM}^{(\mathrm{LT})}(\boldsymbol{le}_i^{k-1}, \boldsymbol{th}_i^{k-1}) \tag{2}$$

Similarly, we have $1 \leq i \leq n$ when t_k is a token-level task, and let $i=0$ when t_k is a sentence-level task. It means that Equation 2 can be used for token-level or sentence-level transfer.

Encoders. Then the transferred label will be fed to encoders. As shown in Figure 4, the inputs of Encoder$^{(k)}$ include three parts: the token embeddings, the transferred label, and the outputs of $(k-1)$th encoder. And the outputs $\boldsymbol{H}^k$ can be represented by:

$$\boldsymbol{H}^k = \text{Encoder}^{(k)}\left(\boldsymbol{E}, \boldsymbol{TH}^{k-1}, \boldsymbol{H}^{k-1}\right) \tag{3}$$

Generally, the choice of encoder falls into three categories: RNN, CNN, and Transformer (Vaswani et al., 2017). And we mainly implement RNN and CNN in our model because the memory complexity of Transformer is $\mathrm{O}(n^2)$ (Kitaev et al., 2020), which is much higher for long text.

For the MTL scenarios JERE and ABSA, the encoders are based on bidirectional LSTM (BiLSTM). Equation 3 becomes:

$$\boldsymbol{H}^k = \text{BiLSTM}^{(k)}\left(\boldsymbol{E} \oplus \boldsymbol{TH}^{k-1} \oplus \boldsymbol{H}^{k-1}\right) \tag{4}$$

where $\oplus$ denotes the concatenation operation along the last dimension. And $\boldsymbol{H}^k$ is the feature representation of X for task t_k with shape $n \times d_h$, and d_h is the size of hidden states.

For the MTL scenario LJP, the subtasks are all sentence-level classification tasks and we empirically find that involving CNN performs better than simply adopting BiLSTM (see Section 5.4). And we employ CNN encoders (Kim, 2014) followed with max pooling to generate initial sentence-level representations $\boldsymbol{h}_{init}^k$ and a shared LSTM layer to generate final representations $\boldsymbol{h}_0^k$ for task t_k:

$$\boldsymbol{h}_{init}^k = \text{pool}\left(\text{CNN}^{(k)}\left(\boldsymbol{E}\right)\right) \tag{5}$$

$$\boldsymbol{h}_0^k = \overrightarrow{LSTM}^{(\mathrm{s})}\left(\boldsymbol{h}_{init}^k \oplus \boldsymbol{th}_0^{k-1},\ \boldsymbol{h}_0^{k-1}\right) \tag{6}$$

where $\boldsymbol{th}_0^{k-1}$ denotes the sentence-level transferred label embedding.

Predictors. Then the predictors will process $\boldsymbol{H}^k$ (or $\boldsymbol{h}_0^k$) and $\boldsymbol{TH}^{k-1}$ (or $\boldsymbol{th}_0^{k-1}$) as follows:

$$\widehat{\boldsymbol{y}}_i^k = \text{Predictor}^{(k)}\left(\boldsymbol{h}_i^k \oplus \boldsymbol{th}_i^{k-1}\right) \tag{7}$$

where $1 \leq i \leq n$ for JERE and ABSA, and $i=0$ for LJP. And $\widehat{\boldsymbol{y}}_i^k$ is the predicted probability distribution with shape for the categories in $\mathcal{C}_k$. The predictors are simply based on fully connected networks, and sequence labeling tasks do not involve conditional random field (CRF) (Lafferty et al., 2001). More details can be found in Appendix.

4.4 Gumbel Sampling

During the training stage, for task t_k, we can pre-train the network using the gold labels $Y^j = \{y_i^j\}_{j<k}$ of all the low-level tasks. However, the train-test discrepancy has not been tackled because the model uses predicted labels of low-level tasks during testing. To deal with the problem, we use GS to sample a label from the predicted probability distribution $\widehat{\boldsymbol{y}}_i^j$. Specifically, we assume that Predictor$^{(j)}$ involves a logit value $\boldsymbol{o}_i^j$ followed with a softmax function to produce the probability distribution $\widehat{\boldsymbol{y}}_i^j$.

$$\widehat{\boldsymbol{y}}_i^j = \text{softmax}(\boldsymbol{o}_i^j) \tag{8}$$

Gumbel-softmax uses a re-parameter trick to approximate the multinomial sampling by:

$$\widetilde{\boldsymbol{y}}_i^j = \text{softmax}((\boldsymbol{o}_i^j + \mathbf{g})/\tau) \tag{9}$$

where $\mathbf{g}$ samples from Gumbel$(0, 1)$ and τ is the temperature. When $\tau \rightarrow 0$, $\widetilde{\boldsymbol{y}}_i^j$ approximated to the one-hot vector of a sampled value from $\widehat{\boldsymbol{y}}_i^j$. During training, our model will replace gold labels by $\widetilde{\boldsymbol{y}}_i^j$. And then a low-level task will have certain probabilities to sample a counterfactual value, and get feedback from high-level tasks if the causal effect is actually existed.

4.5 Interpretation From a Causal Perspective

After training, we attempt to interpret our model from a causal perspective to inspect what matters for predictions. The key idea is causal effect estimation from observed data (Veitch et al., 2019), which is based on Pearl's theory with the intervening operation (Pearl, 2010). Considering two tasks t_m and t_n, we would estimate the causal effect of a label y_i^m of task t_m on a label y_j^n of task t_n. We first intervene y_i^m to get a random counterfactual value $\overline{y}_i^m \neq y_i^m$. Under the mediation assumption, the average causal effect is estimated by:

$$\begin{aligned}\Psi\left(y_i^m, y_j^n\right) = \mathbb{E}\left[f\left(y_j^n | y_i^m, H^n(y_i^m)\right)\right] \\ - \mathbb{E}\left[f\left(y_j^n | \overline{y}_i^m, H^n(\overline{y}_i^m)\right)\right]\end{aligned} \tag{10}$$

where $\mathbb{E}(\cdot)$ stands for the expectation operation on the observed data, and $f(y_j^n | \overline{y}_i^m, H^n(\overline{y}_i^m)$ stands for the predicted probability of y_j^n given $\overline{y}_i^m$ and the corresponding features $H^n(\overline{y}_i^m)$ for task t_n.

Besides estimating the causal effect of labels, we also inspect the influence of the elements in X like n-grams. We can intervene the original sequence to get another text sequence $\overline{X}_{x_g}$ with an n-gram x_g masked out. Since n-grams may be quite sparse, only the individual causal effect is estimated:

$$\psi\left(x_g, y_j^n\right) = f^n\left(y_j^n | X\right) - f^n\left(y_j^n | \overline{X}_{x_g}\right) \tag{11}$$

where $f^n(\cdot)$ represents the prediction for task t^n given a text sequence.

Dataset	Train	Dev	Test	Type
JERE	351 7,273	80 1,765	80 1,535	Document Sentence
ABSA	2,587	457	800	Sentence
LJP	102,177	13,143	25,149	Document

Table 1: Statistics of three MTL datasets.

5 Experiments

5.1 Datasets

To verify the effectiveness of our model, we experiment on three datasets corresponding to three MTL scenarios. For JERE, we use the ACE05 corpus (Doddington et al., 2004), which covers 7 types of entities and 6 types of relations. We use the same data splits as previous work (Katiyar and Cardie, 2017; Sanh et al., 2018). For ABSA, we use restaurant domain reviews of SemEval-2014 task 4 (Pontiki et al., 2014). ATE is a simple BIO tagging task, and ACD is a multi-label classification task with 5 categories. Furthermore, we randomly hold-out 15% of the training data as the development set. For LJP, we use the CAIL (Chinese AI and Law Challenge) 2018 dataset. Following (Zhong et al., 2018; Yang et al., 2019), we only consider those cases with a single law article and single charge. Meanwhile, those infrequent law articles and charges (less than 100 in the train set) are not included. And we divide the terms into non-overlapping intervals, which is consistent with (Zhong et al., 2018). The number of categories for RAP, CP, and PTP is 94, 116, and 11, respectively. The statistics of the filtered datasets can be found in Table 1.

5.2 Baselines

The compared models include two single-task models which use **BiLSTM** and **CNN** (Kim, 2014) as encoders respectively, and three conventional multi-task models including **FSMTL** (Liu et al., 2017; Zheng et al., 2018), **SPMTL** (Liu et al., 2017; Zheng et al., 2018), and **HMTL** (Sanh et al., 2018). Besides these models, we also compare several SOTA models for each MTL scenario, which will be cited in the following subsections.

5.3 Evaluation and Settings

The evaluation metrics are micro Precision (P), Recall (R), and F_1 scores for each subtask in JERE and ABSA. For LJP, the evaluation metrics are accuracy (Acc.), macro P, macro R and macro F_1 scores which are consistent with (Zhong et al., 2018; Yang et al., 2019).

For all models, we use the Allennlp framework to build neural networks (Gardner et al., 2018). The hidden size of BiLSTM and label embeddings is 300. We also investigate each model with BERT-large-uncased (Devlin et al., 2019) or 300-dimensional Glove (Pennington et al., 2014) for JERE and ABSA. For LJP, since it is a Chinese dataset, we use THULAC (Maosong et al., 2016) for word segmentation. We randomly initialize the

http://cail.cipsc.org.cn/index.html

	Tasks	EMD			RE			ATE			ACD		
	Metrics	P	R	F$_1$	P	R	F$_1$	P	R	F$_1$	P	R	F$_1$
STL	BiLSTM	80.60	72.55	76.36	52.89	35.49	42.48	75.99	70.90	73.36	85.88	83.71	84.78
	BiLSTM + BERT	86.36	87.66	87.01	66.86	53.35	59.35	85.08	88.01	86.52	89.80	87.61	88.69
MTL	FSMTL	73.85	75.02	74.43	54.06	25.03	34.22	77.80	72.93	75.29	76.63	50.54	60.90
	SPMTL	80.39	71.61	75.75	60.17	40.66	48.53	75.92	74.51	75.21	89.10	83.71	86.32
	HMTL	63.98	74.91	69.01	56.15	39.13	46.12	69.59	79.72	74.31	89.04	83.22	86.03
	FSMTL + BERT	87.09	86.56	86.82	54.54	29.61	38.38	85.19	86.77	85.98	79.46	48.68	60.38
	SPMTL + BERT	86.52	87.45	86.98	56.69	61.69	59.09	85.04	**89.77**	87.34	89.42	88.20	88.80
	HMTL + BERT	86.82	87.33	87.07	64.42	58.52	61.33	85.34	89.33	87.29	90.01	87.02	88.49
SOTA for JERE	(Sanh et al., 2018) (With CR) *	87.15	87.33	87.24	**70.40**	56.40	62.69	-	-	-	-	-	-
	(Dixit and Al-Onaizan, 2019)	85.85	86.10	85.98	68.02	58.38	62.83	-	-	-	-	-	-
	(Luan et al., 2019) (With CR) *	-	-	**88.40**	-	-	63.20	-	-	-	-	-	-
SOTA for ABSA	(Luo et al., 2019a)	-	-	-	-	-	-	-	-	85.31	-	-	-
	(Xue et al., 2017)	-	-	-	-	-	-	-	-	83.65	-	-	88.91
	(Movahedi et al., 2019)	-	-	-	-	-	-	-	-	-	91.60	**89.63**	90.61
Ours	CMTL	79.84	71.39	75.38	55.48	44.07	49.12	80.27	78.92	79.59	89.08	85.96	87.49
	CMTL + BERT	**88.29**	**87.53**	87.91	69.45	**62.51**	**65.80**	**87.24**	88.62	**87.93**	**93.24**	88.88	**91.00**
Gold Labels	CMTL + BERT (Gold EMD)	-	-	-	69.63	67.10	68.34	-	-	-	-	-	-
	CMTL + BERT (Gold ATE)	-	-	-	-	-	-	-	-	-	92.26	90.73	91.49

Table 2: Experiment results of different models for JERE and ABSA. The results marked with (*) means that their models use an additional task, Coreference Resolution (CR). Note that previous SOTA models are task-specific, which means that the SOTA models for JERE (or ABSA) are not ready for ABSA (or JERE). The rows with "Gold Labels" means using gold labels of low-level tasks.

	Tasks	RAP				CP				PTP			
	Metrics	Acc.	P	R	F$_1$	Acc.	P	R	F$_1$	Acc.	P	R	F$_1$
STL	BiLSTM	84.94	84.25	81.17	81.98	84.06	83.72	81.22	81.59	38.44	36.32	32.38	32.93
	CNN	86.49	86.67	83.43	84.23	86.95	86.14	85.01	84.95	38.80	35.13	34.42	34.06
MTL	FSMTL-CNN	86.47	85.50	83.30	83.64	86.89	86.52	84.56	84.87	38.68	36.98	34.12	35.08
	SPMTL-CNN	86.42	86.31	82.84	83.82	87.50	87.43	84.18	85.03	39.09	37.58	33.89	35.04
	HMTL-CNN	87.17	87.04	83.58	84.54	86.64	85.92	84.76	84.63	38.61	36.03	35.02	35.27
SOTA	(Zhong et al., 2018)	85.95	86.63	83.18	84.10	85.47	84.72	83.58	83.37	38.45	38.48	34.71	35.63
	(Yang et al., 2019)	85.82	85.87	82.66	83.63	85.39	84.46	83.31	83.09	37.11	38.50	33.96	35.29
Ours	CMTL	**86.85**	**87.20**	**84.09**	**84.93**	**87.64**	**87.53**	**85.08**	**85.69**	**39.61**	**40.70**	**35.50**	**37.35**
Gold Labels	CMTL (Gold RAP)	-	-	-	-	97.68	96.56	97.04	96.63	41.71	40.14	37.38	38.15
	CMTL (Gold RAP, CP)	-	-	-	-	-	-	-	-	41.86	40.39	37.58	38.37

Table 3: Experiment results of different models for LJP. The rows with "Gold Labels" means using gold labels of low-level tasks.

word embeddings. For CNN-based models, we set the number of filters as 512 and the sliding window length as 2,3,4,5 (each window contains 128 filters). The temperature of GS τ is set to 0.05. The batch size is 32, and the learning rate is 5×10^{-4}. The maximum training epochs for JERE, ABSA, and LJP are 80, 20, and 10 respectively, and each model will stop training when F$_1$ scores reach the lowest on the development set in past 10 epochs (the patience is set to 10). Moreover, a special setting for LJP is that we pre-train our model for 5 epochs with gold labels and train the next 5 epochs with GS. For the other two MTL scenarios that have fewer categories of labels, pre-training is not involved. We report the averaged metrics after the training process is repeated 5 times.

5.4 Main Results

We first present the experiment results on two English datasets for JERE and ABSA, respectively, in Table 2. Regarding JERE, our model achieves a SOTA result on RE and beats the model proposed by (Luan et al., 2019), which uses an additional task (coreference resolution, CR) by 2.60 F$_1$ points. Among the models without CR, our model achieves the best results on both EMD and RE, improving F$_1$ scores by 1.93 and 2.97 points, respectively. Regarding ABSA, our model achieves new SOTA results on ATE and ACD, which increases F$_1$ scores by 2.62 and 0.39 points, respectively. Table 2 also shows the results of conventional MTL models, including FSMTL, SPMTL, and HMTL. Our model consistently outperforms them when using or not using BERT embeddings.

Then we present the experiment results on the Chinese dataset for LJP in Table 3. Note that "HMTL-CNN" is not merely replacing the BiLSTM encoders of "HMTL" by CNN, because we empirically find this way does not perform well. Therefore, we denote the ablated model of our model as "HMTL-CNN" which is consistent with the scenarios JERE and ABSA. It is also worth noting that the previous SOTA models for LJP are re-implemented by us because we get slightly different data splits after preprocessing the dataset. Moreover, they did not utilize the development set and only tested their models after training certain epochs. As a result, some models may greatly suffer from overfitting at the final epoch. In our experiments, as shown in Table 3, previous SOTA models perform best on PTP, and our model further improves the F_1 scores by 1.72 points over the best of them. Compared with other baseline models, our model performs best on all subtasks.

We also show the results of a toy experiment where our model uses gold labels of low-level tasks in Table 2 and 3. In the three MTL scenarios, using gold labels of low-level tasks leads to performance gains to high-level tasks. The results confirm the existence of label dependencies between tasks. It means that if humans rectify the predictions of low-level tasks, our model can utilize them to improve the predictions of high-level tasks. And conventional MTL models can not utilize this information because they assume the labels to be conditional independent.

Tasks	EMD	RE	ATE	ACD	RAP	CP	PTP
Metrics	F_1	F_1	F_1	F_1	F_1	F_1	F_1
CMTL	**87.91**	**65.80**	**87.93**	**91.00**	**84.93**	**85.69**	**37.35**
HMTL+LT	87.57	63.56	87.09	89.69	84.01	84.67	35.61
HMTL	87.07	61.33	87.29	88.49	84.54	84.63	35.27
CMTL (Indirect)	87.74	65.66	87.66	90.86	84.84	85.28	37.21
CMTL (Direct)	87.11	64.09	87.89	90.52	84.51	84.81	37.02

Table 4: Ablation analysis. The mechanisms GS and LT are eliminated from CMTL one after another, resulting in models "HMTL+LT" and "HMTL", respectively. Furthermore, only keeping the indirect or the direct causal path of LT results in models "CMTL (Indirect)" and "CMTL (Direct)", respectively.

5.5 Ablation Study

To analyze which mechanisms are driving the improvements, we present the results of an ablation study in Table 4. We first eliminate GS and LT from CMTL one after another, which results in models "HMTL+LT" and "HMTL". As shown, GS and LT are both influential, especially for high-level tasks. For example, eliminating GS leads to a drop of F_1 score by 2.24 points on RE, and eliminating the two mechanisms leads to a significant drop of 4.47 points. Moreover, we only keep the indirect and the direct causal path of CMTL, which results in models "CMTL (Indirect)" and "CMTL (Direct)" respectively. Both the two ablated models perform slightly worse than CMTL. Moreover, the indirect causal path is more important than the direct one for most subtasks.

Case 1 for JERE:
*Three people were killed and about 50 injured when another Briton, **Asif Muhammad Hanif**, 21, from **London**, detonated explosives strapped to his body.*

EMD: **B-PER, I-PER, L-PER**——**U-LOC**

RE: HMTL: None (×) CMTL: GEN-AFF 133 (√)

Case 2 for LJP:
The defendant Jiang and others went to the residence of the victim Zhong. ... During this period, Jiang and others snatched 5,000 yuan of cash and bank card from Zhong's wallet, and forced Zhong to tell the bank card password. ... After succeeding, Jiang and others sent Zhong to Huangjiang Town and fled with the money.

	Relavant Article	Charge	Prison Term
HMTL	238 (√)	Kidnapping (×)	24-36 Months (×)
CMTL	238 (√)	Illegal Detention (√)	12-24 Months (√)

Article 238: *Those who **illegally detain** others or deprive others of their personal freedom shall be sentenced to fixed-term imprisonment, which is not more than three years.*

Figure 5: Example cases with the prediction results of HMTL and CMTL. The symbol $\surd$ denotes a correct prediction and $\times$ denotes a wrong prediction.

5.6 Case Study

Influence of Label Transfer. Generally, LT enables a high-level task to utilize all its lower-level tasks' predictions and, therefore, improves the consistency of the predicted results. To directly see the influence, we give some cases in Figure 5. For example, in Case 1, both HMTL and CMTL successfully recognize the entities, including "Asif Muhammad Hanif" and "London". Nevertheless, HMTL does not correctly predict their relationship "GEN-AFF" (citizens and the place they come from) while CMTL correctly predicts it. Another example is Case 2, which shows the translated document of a Chinese legal case. As shown, the relevant law is Article 238, which describes the crime of illegal detention. But the predicted charge of HMTL is kidnapping, which is a more serious crime. These inconsistent judgments are unacceptable to the judge or the public in practice.

	Illegal Detention	Illegal Trade of Drug-Making	Causing Traffic Accident	Dangerous Driving
Article 238	0.30	-	-	-
Article 350	-	0.90	-	-
Article 133	-	-	0.07	0.04

Table 5: The estimated causal effects of several law articles on certain charges in our model for LJP. The symbol "-" represents that the estimated effect is below 0.001.

Estimated Causal Effect. To show that the estimated causal effect (computed by Equation 10) in our model is consistent with humans' prior knowledge, we present some results in Table 5. As shown, *Article 238* has a causal effect on *Illegal Detention* with 0.30 points, and has no effect on other charges. This is consistent with legal knowledge (view *Article 238* in Figure 5). And *Article 350* has a causal effect on *Causing Traffic Accident* with 0.90 points that are quite high. The reason may be that *Article 350* has only 163 samples in the train set, while *Article 238* has 1,427 samples. The confidence of infrequent labels can be greatly improved by knowing the low-level gold labels. The third row shows the kind of one-to-many causal effect that *Article 133* has a causal effect on both *Causing Traffic Accident* and *Dangerous Driving*. But the effect is small, and the prediction should mainly count on the features extracted from the input text.

Case 3 for LJP:
被告人张某驾驶安全设施不合格的重型罐式半挂牵引车，与沿庞庄路由西向东行驶的被害人张某某驾驶的无牌二轮摩托车相撞，致被害人张某某受伤，经医院抢救无效死亡。被告人张某负事故主要责任。
The defendant, Zhang, was driving a heavy tank tractor with unqualified safety facilities, and collided with an unlicensed two-wheeled motorcycle driven by the victim Zhang,... The victim Zhang was injured and ***died after being rescued by the hospital*** **(N_1)***. The defendant Zhang was* ***mainly responsible for the accident*** **(N_2)**.
Prediction: HMTL+LT: Article 233 (×) CMTL: Article 133 (√)

	Influential 4-grams	Article 133	Article 233
HMTL+LT	医院, 抢救, 无效, 死亡 (N_1)	-0.47	0.48
	负, 事故, 主要, 责任 (N_2)	0.45	-0.45
CMTL	医院, 抢救, 无效, 死亡 (N_1)	-0.02	0.02
	负, 事故, 主要, 责任 (N_2)	0.80	-0.79

Figure 6: The estimated causal effects of 4-grams. Article 133 is about dangerous driving, and Article 233 is about negligently causing one's death. Although the two law articles both describe one's death, Article 133 has priority in the event of traffic accidents.

Influence of Gumbel Sampling. GS enables a low-level task to get useful feedback from its higher-level tasks. To see the influence, we show another case for task RAP (the lowest-level task in LJP) in Figure 6. As shown, the gold label of the case is *Article 133*, which is about dangerous driving. Without GS, the model HMTL+LT predicts an incorrect result, *Article 233*, about negligently causing one's death. The two law articles both describe one's death, but *Article 133* has priority in the event of traffic accidents. Figure 6 shows the estimated causal effect of each 4-gram (computed by Equation 11). CMTL captures the priority of *Article 133* by understanding that the translated n-gram "mainly responsible for the accident" is more important (with a causal effect of 0.80 on *Article 133*) than the n-gram "died after being rescued by the hospital" (with a causal effect of -0.02).

6 Conclusion

In this paper, we investigated the MTL problem with logically dependent tasks. We first analyze MTL models from the perspective of causal inference and then propose a model CMTL to utilize task dependencies properly. The model achieves SOTA results on 6 out of 7 subtasks and improves the consistency of predicted results of different subtasks. In the future, we are interested in social science topics, such as modeling the causal effect between mental health and the suicide decisions reflected through social media, which may help predict and stop the final decisions.

Acknowledgments

The authors would like to thank the anonymous reviewers for their constructive comments. This research was supported by National Key Research and Development Program of China under Grant 2018YFC0830400.

References

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems With Applications*, pages 34–45.

Rich Caruana. 1997. Multitask Learning. *Machine Learning*, 28(1):41–75.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL*, pages 4171–4186.

Kalpit Dixit and Yaser Al-Onaizan. 2019. Span-Level Model for Relation Extraction. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5308–5314.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. *LREC*, 2:1.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew E. Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. *CoRR*, abs/1803.07640.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-Predict: Parallel Decoding of Conditional Masked Language Models. *EMNLP*.

Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. 2018. Long Text Generation via Adversarial Training with Leaked Information. *NAACL*, pages 5141–5148.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple nlp tasks. *EMNLP*, pages 1923–1933.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An Interactive Multi-Task Learning Network for End-to-End Aspect-Based Sentiment Analysis. *ACL*, pages 504–515.

S Hochreiter and J Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735.

Arzoo Katiyar and Claire Cardie. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. *ACL*, 1:917–928.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *arXiv:1408.5882 [cs]*. ArXiv: 1408.5882.

Nikita Kitaev, ukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The Efficient Transformer. *ICLR*.

Matt J. Kusner and José Miguel Hernández-Lobato. 2016. GANS for Sequences of Discrete Elements with the Gumbel-softmax Distribution. *arXiv:1611.04051*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML*.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1:402–412.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A Unified Model for Opinion Target Extraction and Target Sentiment Prediction. *national conference on artificial intelligence*, pages 6714–6721.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial Multi-task Learning for Text Classification. *Meeting of the Association for Computational Linguistics*, pages 1–10.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-Task Deep Neural Networks for Natural Language Understanding. *meeting of the association for computational linguistics*, pages 4487–4496.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A General Framework for Information Extraction using Dynamic Span Graphs. *north american chapter of the association for computational linguistics*, pages 2036–3046.

Huaishao Luo, Tianrui Li, Bing Liu, Bin Wang, and Herwig Unger. 2019a. Improving Aspect Term Extraction with Bidirectional Dependency Tree Representation. *IEEE Transactions on Audio, Speech, and Language Processing*, pages 1201–1212.

Huaishao Luo, Tianrui Li, Bing Liu, and Junbo Zhang. 2019b. DOER: Dual Cross-Shared RNN for Aspect Term-Polarity Co-Extraction. *meeting of the association for computational linguistics*, pages 591–601.

Sun Maosong, Chen Xinxiong, Zhang Kaixu, Guo Zhipeng, and Liu Zhiyuan. 2016. Thulac: An efficient lexical analyzer for chinese. *Technical report*.

Sajad Movahedi, Erfan Ghadery, Heshaam Faili, and Azadeh Shakery. 2019. Aspect Category Detection via Topic-Attention Network. *arXiv:1901.01183 [cs]*. ArXiv: 1901.01183.

Weili Nie, Nina Narodytska, and Ankit Patel. 2019. RelGAN: Relational Generative Adversarial Networks for Text Generation. *international conference on learning representations*.

Judea Pearl. 2010. On measurement bias in causal inference. *uncertainty in artificial intelligence*, pages 425–432.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. *EMNLP*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *north american chapter of the association for computational linguistics*.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. *Proceedings of the*

8th International Workshop on Semantic Evaluation (SemEval 2014), pages 27–35.

Marcaurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. *ICLR*.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2017. Learning what to share between loosely related tasks. *arXiv preprint arXiv:1705.08142*.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2018. A Hierarchical Multi-task Approach for Learning Embeddings from Semantic Tasks. *arXiv:1811.06031 [cs]*. ArXiv: 1811.06031.

Bernhard Schölkopf. 2019. Causality for Machine Learning. *arXiv:1911.10500 [cs, stat]*. ArXiv: 1911.10500.

Anders Sgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *neural information processing systems*, pages 5998–6008.

Victor Veitch, Dhanya Sridhar, and David M Blei. 2019. Using text embeddings for causal inference. *arXiv: Learning*.

Wei Xue, Wubai Zhou, Tao Li, and Qing Wang. 2017. MTNA: A Neural Multi-task Model for Aspect Category Classification and Aspect Term Extraction On Restaurant Reviews. *IJCNLP*.

Wenmian Yang, Weijia Jia, XIaojie Zhou, and Yutao Luo. 2019. Legal Judgment Prediction via Multi-Perspective Bi-Feedback Network. *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 4085–4091. ArXiv: 1905.03969.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. *NAACL*, pages 2852–2858.

Renjie Zheng, Junkun Chen, and Xipeng Qiu. 2018. Same Representation, Different Attentions: Shareable Sentence Representation Learning from Multiple Tasks. *international joint conference on artificial intelligence*, pages 4616–4622.

Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Chaojun Xiao, Zhiyuan Liu, and Maosong Sun. 2018. Legal Judgment Prediction via Topological Learning. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3540–3549.

A Details of Predictors and Loss Function

The subtasks in three MTL scenarios can be categorized into four types: sequence labeling, inter-token multi-label classification, multi-label text classification, and single-label text classification. Since the primary goal of our work is to investigate the task dependency, the architectures of predictors are based on simple fully-connected neural networks (FCNNs).

- If the task t_k is EMD or ATE which belongs to **sequence labeling**, given feature representations $\boldsymbol{H}^k=\{\boldsymbol{h}_i^k\}, 1\leq i\leq n$, the prediction layer will make a token-level prediction as follows:

$$\widehat{\boldsymbol{y}}_i^k = \mathrm{softmax}(\boldsymbol{W}_p^k \boldsymbol{h}_i^k + \boldsymbol{b}_p^k) \tag{12}$$

where $\boldsymbol{W}_p^k$ and $\boldsymbol{b}_p^k$ are the trainable parameters of the FCNN. The loss is computed by cross-entropy:

$$\mathcal{L}^k = -\sum_{i=1}^{n}\sum_{c=1}^{|\mathcal{C}_k|} y_{i,c}^k \log \widehat{y}_{i,c}^k \tag{13}$$

where $y_{i,c}^k$ denotes the ground-truth value of the cth category for ith token, and $\mathcal{C}_k$ is the set of the categories.

- If task t_k is RE which belongs to **inter-token multi-label classification**, and the prediction process can be described by:

$$\widehat{\boldsymbol{y}}_{i,j}^k = \sigma(\boldsymbol{V}_p^k f\left(\boldsymbol{U}_p^k \boldsymbol{h}_i^k + \boldsymbol{W}_p^k \boldsymbol{h}_j^k + \boldsymbol{b}_p^k\right)) \tag{14}$$

where $\widehat{\boldsymbol{y}}_{i,j}^k$ denotes the predicted probability distribution of the relations between the ith and jth tokens. And $\boldsymbol{V}_p^k$, $\boldsymbol{U}_p^k$, $\boldsymbol{W}_p^k$ and $\boldsymbol{b}_p^k$ are the trainable parameters of the FCNN. The symbol $\sigma(\cdot)$ stands for the sigmoid function, and $f(\cdot)$ stands for the element-wise activation function ($relu$ in this paper). The loss $\mathcal{L}_{\text{re}}$ is computed according to cross-entropy:

$$\mathcal{L}^k = -\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{c=1}^{|\mathcal{C}_k|} y_{i,j,c}^k \log \widehat{y}_{i,j,c}^k \tag{15}$$

where $y_{i,j,c}^k$ denotes the ground-truth value of the cth category for the relation between the ith and jth tokens. Note that we only consider relations between the last token of the head entity mentions. Redundant relations are therefore not classified.

- If task t_k is ACD which belongs to **multi-label text classification**, the prediction process is as follows:

$$\boldsymbol{h}_0^k = \text{pool}(\boldsymbol{H}^k) \tag{16}$$

where $\boldsymbol{h}_0^k$ represents the sentence-level feature representation obtained by the max-pooling function pool$(\cdot)$ over the tokens. And the predicted probability distribution is:

$$\widehat{\boldsymbol{y}}^k = \sigma(\boldsymbol{W}_p^k \boldsymbol{h}_0^k + \boldsymbol{b}_p^k) \tag{17}$$

Then the loss is computed by cross-entropy as follows:

$$\mathcal{L}^k = \sum_{c=1}^{|\mathcal{C}_k|} y_c^k \log \widehat{y}_c^k \tag{18}$$

- If task t_k is RAP, CP, or PTP which belongs to **single-label text classification**, the prediction process is as follows:

$$\widehat{\boldsymbol{y}}^k = \text{softmax}(\boldsymbol{W}_p^k \boldsymbol{h}_0^k + \boldsymbol{b}_p^k) \tag{19}$$

where the difference from multi-label text classification tasks is the use of softmax$(\cdot)$ instead of $\sigma(\cdot)$. And the loss is also computed by cross-entropy:

$$\mathcal{L}^k = \sum_{c=1}^{|\mathcal{C}_k|} y_c^k \log \widehat{y}_c^k \tag{20}$$

When considering multi-task learning, we denote the set of tasks by $\mathcal{T} = \{t_1, t_2, ..., t_k, ..., t_K\}$ and then sum up the losses of tasks by:

$$\mathcal{L}^{\text{MT}} = \sum_{k=1}^{K} \lambda^k \mathcal{L}^k \tag{21}$$

where λ^k is the hyper-parameter for each task t_k. We empirically set $\lambda^k = 1.0$ in this paper.

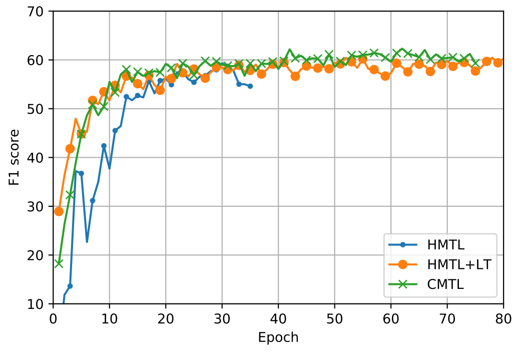
B Validation Performance During Training

We also show the validation performance of several models on the highest-level subtask for three MTL scenarios during training in Figure 7. As shown, the F_1 scores and losses on the development set of three models are presented, including CMTL and two ablated models, HMTL+LT and HMTL.

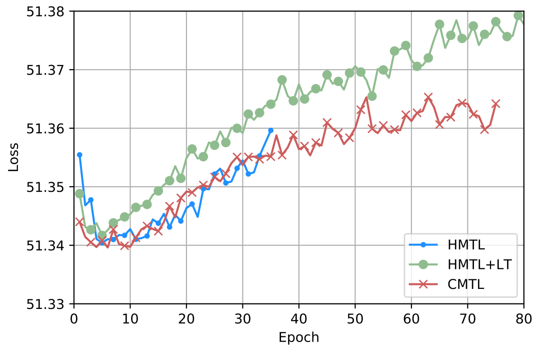
For RE, HMTL ran out of patience at about 35 epoch as it reached the lowest F_1 score in the past 10 epochs. And HMTL+LT and CMTL kept training for nearly 80 epochs. The best F_1 score of CMTL was slightly higher than that of HMTL+LT by 1.86 points, and the loss curve was more stable. Similarly, for ACD and PTP, the best F_1 scores of CMTL were consistently higher than the other two models, and the loss of CMTL was relatively stable. These results demonstrate that our model can better utilize the task dependencies and be more robust than the other two ablated models.

Moreover, an interesting result was that the validation loss of HMTL+LT grew faster than HMTL. The reason may be that the predicted labels of low-level tasks in HMTL+LT excessively influenced the decision of high-level tasks, leading to cascading errors. If an incorrectly predicted label of the low-level task is fed, the high-level task will have high confidence to make a wrong prediction, making the loss of HMTL+LT larger than HMTL. By adding Gumbel sampling, our model achieved the smallest loss on the development set, which indicated that Gumbel sampling properly considered the causal effect and alleviated the cascading errors.

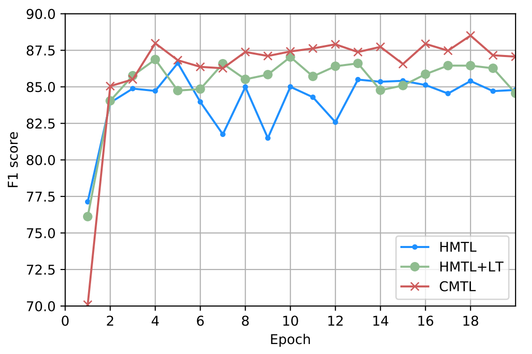
Each model was trained on a Tesla P100 with a maximum memory of 16GB.



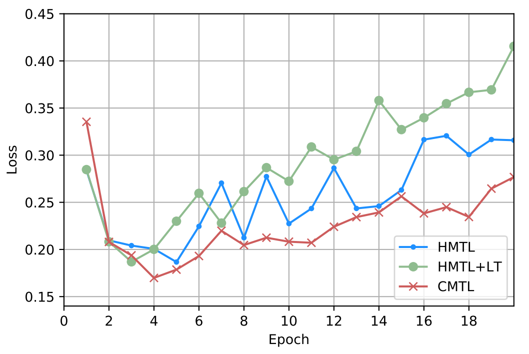
(a) F_1 scores on the development set for RE in JERE.



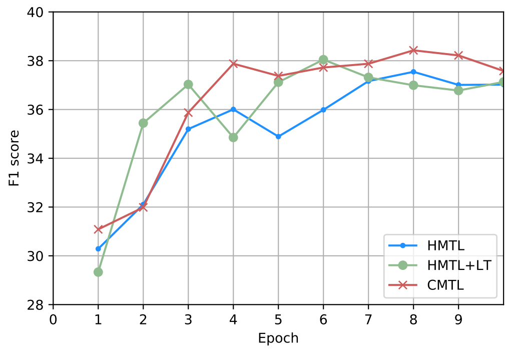
(b) Loss on the development set for RE in JERE.



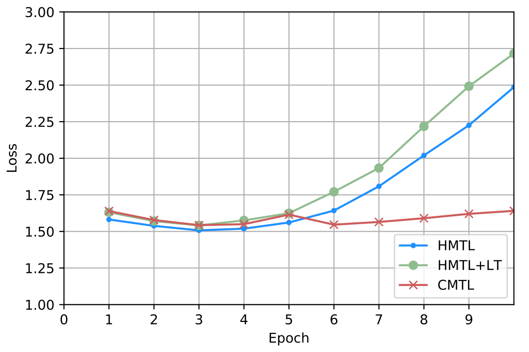
(c) F_1 scores on the development set for ACD in ABSA.



(d) Loss on the development set for ACD in ABSA.



(e) F_1 scores on the development set for PTP in LJP.



(f) Loss on the development set for PTP in LJP.

Figure 7: The validation performance of the highest-level subtask in three MTL scenarios.