SRCB at the NTCIR-16 Real-MedNLP Task

Yongwei Zhang
Ricoh Software Research Center (Beijing) Co., Ltd.
Yongwei.Zhang@cn.ricoh.com

Rui Cheng
Ricoh Software Research Center (Beijing) Co., Ltd.
Rui.Cheng@cn.ricoh.com

Lu Luo
Ricoh Software Research Center (Beijing) Co., Ltd.
Lu.Luo@cn.ricoh.com

Haifeng Gao
Ricoh Software Research Center (Beijing) Co., Ltd.
Haifeng.Gao@srcb.ricoh.com

Shanshan Jiang
Ricoh Software Research Center (Beijing) Co., Ltd.
Shanshan.Jiang@cn.ricoh.com

Bin Dong
Ricoh Software Research Center (Beijing) Co., Ltd.
Bin.Dong@cn.ricoh.com

ABSTRACT

The SRCB participated in subtask1: Few-resource Named Entity Recognition (NER) and subtask3: Adverse Drug Event detection (ADE) in NTCIR-16 Real-MedNLP. This paper reports our approach to solve the problem and discusses the official results. For the Few-resource NER subtask, we developed NER systems based on pre-training model, span-based classification and prompt learning. In addition, data augmentation and model ensemble are used to further improve performance. For ADE subtask, we mainly adopted two methods: multi-class classification and prompt learning. We employed a two-stage training strategy to solve the long tail distribution problem and applied transfer learning to improve performance of model.

KEYWORDS

Few-resource, Named Entity Recognition, Adverse Drug Event detection, span-based classification, prompt learning, multi-class classification

TEAM NAME

SRCB

SUBTASKS

Subtask1-CR-EN
Subtask1-RR-EN
Subtask3-CR-EN (ADE)

1 INTRODUCTION

NTCIR-16 Real-MedNLP[1] is a shared task workshop for medical language processing using actual medical documents (case reports and radiology reports). The goal of this task is to promote the development of practical systems that support various medical services. In Real-MedNLP track this year, we are mainly involved in the following subtasks: subtask1: Few-resource NER(Subtask1-CR-EN,Subtask1-RR-EN) and subtask3: ADE(Subtask3-CR-EN).

The Few-resource NER subtask challenges participants to extract important information from the real medical text. In particular, the participants were asked to perform a recognition and classification of 12 medical entity types and various fine-grained attributes. NER is one of the most basic task in natural language processing, and there have been many research progresses. The most mainstream and effective method is fine-tuning based on pre-trained language model (PLM) like BERT[2]. There are many NER methods, such as sequence tagging model[3], pointer network[4] and span-based model[5–7]. To explore the most suitable PLMs and NER methods for subtask1, we first conduct experiments on different combinations of PLMs and NER methods and improved the model based on prompt learning.

Next, in order to further improve the performance of model, we explored data augmentation methods and model integration methods suitable for subtask1. Later, by comparing the Annotation Guidelines[8] and model prediction results, we found that some error prediction results did not conform to the guidelines, so we further summarized some rules according to the guidelines to further improve the performance of subtask1.

For ADE, the goal of this task is to extract adverse drug event (ADE) information from case reports. This subtask is especially designed for MedTxt-CR. Given an input report, the system extracts the ADE information from the report. We considered the task a multi-class classification problem to detect ADE certainty of each given disease or medicine into one class among. Prompt learning was also introduced because it has showed good performance in small sample classification in recent study.

For multi-class classification, we try a two-stage training strategy to solve the problem for long tail distribution data. Furthermore, we create related tasks to improve performance of model by transfer learning. For prompt learning, we design the patterns and verbalizer for model. It is worth mentioning that using position information of entity improves the performance of model. Finally, we explore some data augmentation strategies for small sample problem.

2 RELATED WORK

2.1 Span-based model

Span-based model is using span representations derived from a pre-training model like BERT, word and character embeddings. These representations are then shared across the downstream tasks.

2.2 Long-tailed Distribution

Long-tailed learning can be regarded as a more specific and challenging sub-task within class-imbalanced learning. In comparison, in class-imbalanced learning, the number of classes can be very small and the number of minority data is not necessarily small; while in long-tailed learning, there are a large number of classes and the tail-class samples are often very scarce.

The two simplest basic methods of long tailed distribution are re-sampling and re-weighting. The essence of these methods is to

use the known data set distribution to hack the data distribution in the learning process, that is, reverse weighting, strengthen the learning of tail classes and offset the long tail effect.

Re-sampling in earlier studies included under sampling of head classes and over sampling of tail classes. The most commonly used strategy is class-balanced sampling that each sample has the same probability of being selected, regardless of its class. Kang et al. [9] seek to seeks to train models from class-imbalanced samples.

Re-weighting [10, 11] is mainly reflected in the loss of classification. Unlike re-sampling, because of the flexibility and convenience of loss calculation, many complex tasks, such as object detection and instance segmentation, prefer to use re-weighted loss to solve the long-tailed distribution problem.

2.3 Prompt Learning

Prompt Learning refers to processing the input text information according to a specific template, and reconstructing the task into a form that can make full use of the pre-trained language model. Prompt learning is currently a very effective method for few shot classification. Transform the classification problem into a mask language modeling problem[12].

3 METHODS

In this section, we detail our approach of subtask1 and subtask3 in Section 3.1 and Section 3.2.

3.1 Subtask1

In this section, we consider NER as a language model span classification problem. We first introduce the pre-trained language model in Section 3.1.1, and then detail model structure in Section 3.1.2, and last show model ensembles and data augmentation in Section 3.1.3 and Section 3.1.4, respectively.

3.1.1 Pretrained Language Models.

In this task, we tried 5 different PLMs: BERT, BioBERT[3], Clinical BERT[13], Pubmed BERT[14], and Entity BERT[15]. Their model structure is the same as BERT, the differences are that they were pre-trained using different training data (biomedical or clinical domain data) and adopt different mask strategies (entity-centric masking strategy). See Table 1 for a summary of PLMs.

3.1.2 Span-based and Prompt NER Model.

Traditionally, various approaches for NER have been investigated such as sequence tagging model, pointer network and span-based model. We tried all these methods and improve the span-based method by adding prompt learning[16].

Our span classifier takes an arbitrary candidate span as input. We pre-define a set of entity categories. The span classifier maps the span to a class, none represents spans that do not constitute entities. Let's introduce the span-based model with prompt learning.

For a input sentence X consisting of n tokens $x_1, x_2, ..., x_n$. Let $S = s_1, s_2, ..., s_m$ be all the possible spans in X of up to length L. We first define a prompt template for each span s_i, the template is "s_i is a [MASK] entity". "[MASK]" is a mapping word of entity type. Then Splice the prompt information constructed based on the template to the back of the sentence. The model only needs to predict that the word corresponding to [MASK] is the result we want.

We use a pre-trained language model to obtain contextualized representations x_t for each input token x_t, suppose [CLS](representation of overall sentence) is expressed as e_{cls} and [mask] is expressed as e_{mask}. Then use max-pooling to generate the pre-trained model embeddings of span(e_{span}) in sentence and span($e^{'}_{span}$) in prompt template.

Given the span width l, we look-up a width embedding w_l from a dedicated embedding matrix, which contains a fixed size embedding for each span width 1, 2, ... [17]. These embeddings are learned by back-propagation.

This yields the following span representation(whereas $\oplus$ denotes concatenation):

$$e_s = AvgPooling(e_{span}, e^{'}_{span}) \oplus w_l$$

Finally, we add the representation of "[CLS]" and "[MASK]". The final input to the span classifier is:

$$X^s = e_s \oplus e_{cls} \oplus e_{mask}$$

This input is fed into a softmax classifier:

$$y^s = softmax(W^s \cdot X^s + b^s)$$

which yields a posterior for each entity class

3.1.3 Model Ensembles.

In the model parameter tuning stage, since there is only a training set, we use k-fold cross-validation to make full use of the data. When the model parameter tuning tends to be stable, we select several sets of better parameter combinations, use all the training sets to train the data, and then select several lower models as candidate ensemble models. Because of the different model structures we use, the simplest result weighted average method is used for the final integration.

3.1.4 Data Augmentation.

Data augmentation has been proved to be effective in many domains in artificial intelligence.

We first perform sentence linearization to convert labeled sentences into linear sequences, so that language models can be used to learn the distribution of words and tags in gold data.

Given a sentence, first feed the sequence of tokens into the embedding layer to lookup the corresponding embeddings.

A dropout layer is applied to each token embedding to generate new embeddings. Then feed embeddings into LSTM to produce hidden state at each position. Another dropout layer is applied to hidden states.

Finally, a linear and softmax layer is used to predict the next token in the sequence. Through the above process, a new training corpus is generated.

3.2 Subtask3

In this section, we mainly consider the methods fine-tuning on the pre-trained language model, include multi-class classification method and prompt learning based method. In addition, we also used model ensembles and data augmentation.

3.2.1 Multi-class Classification.

This method mainly consists of three parts: transfer learning, two-stage training.

Table 1: PLMs summary

PLMs	Pre-trained data	Pre-trained from scratch?	features
BERT	Wikipedia and BookCorpus	YES	General-domain (out-domain)
BioBERT	PubMed abstracts and PubMed Central full text articles	NO, based on BERT	Mixed-domain
Clinical BERT	MIMIC-III corpus	NO, based on BERT and BioBERT	Mixed-domain
Pubmed BERT	PubMed abstracts and PubMed Central full text articles	YES	In-domain
Entity BERT	MIMIC-III corpus	NO, based on Pubmed BERT	In-domain Entity-centric masking strategy

Transfer Learning The main goal of transfer learning is to apply the knowledge or patterns learned in a certain field or task to different but related fields or problems. Therefore, we mainly create related but different tasks to make the model learn as much knowledge as possible and transfer to this task.

For medicine and disease binary classification task, We train a classification model to judge whether the text to be predicted is disease or medicine.

The cloze test task is to judge whether the candidate is the correct answer to the masked position (original medicine or disease position). Note that the candidate consists of one correct answer and three wrong answer, the class of wrong answer is different with correct answer. For example, the correct answer is a disease, the three wrong answer is medicine from other case reports.

Two-stage Training In CV, BBN[18] uses Bilateral-Branch Network to solve the problem caused by long-tailed distribution data. Inspired by BBN, we proposed the two-stage training that the whole training stage is divided into two parts. Note that the loss function of first stage is ACSL loss[19], the training steps are 80% of total training steps. The loss function of second stage is weighted cross-entropy (WCE) loss, the training steps are 20% of total training steps.

3.2.2 *Prompt Learning.*

Prompt Learning We first discuss pattern tilization training (PET)[12] for text classification tasks, that is, for some text inputs, $X \in x$ must be mapped from a finite set Y to a single output y. Let m be a mask model (MLM) and t be its tag set and $mask \in T$ is the masked mark. We represent the set of all marker sequences as $T*$. Pet requirements:

Pattern: $P(x)$, which maps the original input to a problem containing one mask token;

Verbalizer: $V(y)$, which maps each output to a single token representing its meaning.

Therefore, the original problem has also been transformed. The probability of obtaining y after giving X has also become the probability that the mask model M predicts $V(y)$ at the mask position of $P(x)$.

Patterns design examples:

* 1. text_b + "And it will " + self.mask + " bring the adverse event." + text_a
* 2. text_a + text_b + "And it will " + self.mask + " bring the adverse event."
* 3. "In this article, there is " + self.mask + "having the adverse event"+ text_a + text_b

Verbalizer design : "0": ["not"]; "1": ["unlikely"]; "2": ["probably"]; "3": ["definitely"]

Position Information The position information refers to the beginning, middle and end of a drug or disease in an article. Because of a case report, different positions also represent different meanings, such as diagnosis or discussion. Therefore, adding this kind of information is effective.

3.2.3 *Data Augmentation.*

We adopt two data augmantation strategies in this task. One is the Back Translation, we translate data into Chinese and then into English through Baidu translation API, so as to obtain more training data.

The other is Feature Cutoff, we randomly erase some feature dimensions in the embedding matrix, the embedding after cutting is named augmented embedding. The main idea is to feed original embedding and augmented embedding into shared encoder to get the original logits and augmented logits, and then compute the KL-Divergence between original logits and augmented logits. Finally, we add KL-Divergence to the cross-entropy (CE) loss as the total loss.

3.2.4 *Model Ensembles.*

In the model ensemble stage, we use 5-fold cross-validation to select the optimal model, and then use the weighted average of the results for model ensemble. In order to further improve the performance of the model, we construct different 5-fold data to obtain the model ensemble results, and another weighted ensemble is used to obtain the finally result .We call this two-stage ensemble. Finally, we also try to ensemble the model trained with all the data.

4 EXPERIMENTS

4.1 Subtask1

We use 5-fold cross-validation to make full use of the data. The performance of our model with different technical points on Subtask1-CR-EN training set are listed in Table 2, Table 3 and Table 4. The performance is the evaluation results of 12 tags and attached attributes. Table 2 shows the comparison of the sequence tagging model on different pre-training models(PLMs). Table 3 shows the fixed pre-training model(Pubmed BERT), the comparison of the results of different model structures, include sequence tagging model,

Pointer network and span-based model with prompt learning(Span + PL). Table 4 shows the performance of model with prompt learning(PL) and data augmentation(DA) and model ensemble(ME). In the results, we conclude that the span-based model with prompt learning and augmentation and model ensemble show significant improvement over our final system.

Table 2: Subtask1-CR-EN training set 5-fold cross-validation average results on pre-training models

PMLs	EntityF(CR)	JointF(CR)
Clinical BERT	59.232	55.128
BioBERT	59.664	55.392
Entity BERT	60.296	55.902
Pubmed BERT	62.284	57.464

Table 3: Subtask1-CR-EN training set 5-fold cross-validation average results on different model

Model	EntityF(CR)	JointF(CR)
Sequence tagging	62.284	57.464
Pointer network	63.176	58.946
Span + PL	64.698	59.712

Table 4: Subtask1-CR-EN training set 5-fold cross-validation average results on different technical points

Model	EntityF(CR)	JointF(CR)
Span + PL(Base)	64.698	59.712
+DA	66.760	62.222
+ME	69.996	64.960

4.2 Subtask3

4.2.1 PLMs.

In this experiment, we use three PLMs as our baseline, Pubmed BERT, Clinical BERT, BioBERT. In Table 5, we can observe that Pubmed BERT has the best result in Subtask3-CR-EN training set 5-fold cross-validation average.

4.2.2 Multi-class Classification.

Table 6 shows the Subtask3-CR-EN training set 5-fold cross-validation average results of multi-class classification with different technical points. We can see that all technical points contribute to the performance of the model to a certain extent, among which two-stage learning and data augmentation greatly improve the performance of model.

Table 5: The Subtask3-CR-EN training set 5-fold cross-validation average results of PLMs

PLM	Baseline F1
Pubmed BERT	34.2
Clinical BERT	33.2
BioBERT	32.0

Table 6: Subtask3-CR-EN training set 5-fold cross-validation average results of multi-class classification

Multi-class Classificatiion	53.7
w/o data augmentation	52.5
& w/o cloze test task (Transfer Learning)	51.7
& replace ACSL with CE in two-stage training	50.6
& w/o two-stage training	47.1
& w/o binary classification task (Transfer Learning)	43.6

Table 7: Subtask3-CR-EN training set 5-fold cross-validation average results of prompt learning

Prompt Learning	47.4
w/o data augmentation	43.0
& w/o position information	41.8
& w/o prompt learning	34.2

4.2.3 Prompt Learning.

Table 7 shows the Subtask3-CR-EN training set 5-fold cross-validation average results of prompt learning. The prompt learning has the highest improvement for all strategies. The data augmentation also greatly improve the performance of the model. Note that it is useful to add position information for prompt learning.

5 SUBMISSIONS

5.1 Subtask1

5.1.1 Subtask1-CR-EN.

For Subtask1-CR-EN test set, we submitted 5 runs for comparison and analysis. Each run uses model ensembles and data augmentation, and the training set is expanded by about one time. Pre-training models is Pubmed Bert. The difference is the model structure and the number of label types.

Subtask1-CR-EN-1: Use span-based model with no prompt learning. 12 labels and attached attributes.

Subtask1-CR-EN-2: Use span-based model with prompt learning and some manual summary rules. 8 labels and attached attributes.

Subtask1-CR-EN-3: Use span-based model with prompt learning. 8 labels and attached attributes.

Subtask1-CR-EN-4: Use sequence tagging model and pointer network model and some manual summary rules. 12 labels and attached attributes.

Subtask1-CR-EN-5: Use sequence tagging model and pointer network model and some manual summary rules. 8 labels and attached attributes.

The performance of our model with different technical points on the Subtask1-CR-EN test set for subtask1 are listed in Table 8.

5.1.2 Subtask1-RR-EN.

For Subtask1-RR-EN test set, we submitted 3 runs for comparison and analysis. Each run uses model ensembles and data augmentation, and the training set is expanded by about one time. Pre-training models is Pubmed Bert.

Subtask1-RR-EN-1: Use span-based model with prompt learning and some manual summary rules.

Table 8: Subtask1-CR-EN results for our submitted runs

Run	EntityAcc	EntityP	EntityR	EntityF	JointAcc	JointP	JointR	JointF
Subtask1-CR-EN-1	85.69	65.09	55.38	59.84	84.14	59.69	50.79	54.88
Subtask1-CR-EN-2	88.27	67.65	59.71	63.43	86.58	62.50	55.16	58.60
Subtask1-CR-EN-3	87.77	68.06	57.59	62.39	86.13	62.80	53.14	57.57
Subtask1-CR-EN-4	88.38	58.17	60.71	59.41	86.47	53.46	55.80	54.60
Subtask1-CR-EN-5	88.64	60.90	59.78	60.33	86.66	55.92	54.89	55.40

Table 9: Subtask1-RR-EN results for our submitted runs

Run	EntityAcc	EntityP	EntityR	EntityF	JointAcc	JointP	JointR	JointF
Subtask1-RR-EN-1	92.23	83.14	82.06	82.60	89.96	79.71	78.67	79.19
Subtask1-RR-EN-2	92.28	83.26	82.06	82.66	89.61	79.31	78.17	78.74
Subtask1-RR-EN-3	92.66	79.90	81.34	80.61	90.37	76.50	77.88	77.19

Table 10: Subtask3-CR-EN (ADE) results for our submitted runs

Method	Entity Level (ADEval)				Report Level
	0	1	2	3	
Subtask3-CR-EN-1 (ADE)	97.25	7.69	-	61.54	57.14
Subtask3-CR-EN-2 (ADE)	97.25	0.00	-	63.41	57.14
Subtask3-CR-EN-3 (ADE)	97.18	0.00	-	61.54	52.63
Subtask3-CR-EN-4 (ADE)	97.02	0.00	-	60.00	57.14
Subtask3-CR-EN-5 (ADE)	97.60	0.00	-	66.67	42.86
Subtask3-CR-EN-6 (ADE)	97.11	9.09	-	54.05	47.06

Subtask1-RR-EN-2: Use span-based model with no prompt learning and some manual summary rules.
Subtask1-RR-EN-3: Use sequence tagging model and pointer network model and some manual summary rules.
The performance of our model with different technical points on the Subtask1-RR-EN test set for subtask1 are listed in Table 9.

5.2 Subtask3

5.2.1 Subtask3-CR-EN (ADE).
For Subtask3-CR-EN (ADE) test set, we submitted 6 runs for comparison and analysis. Each run uses model ensembles and data augmentation, and the training set is expanded by about one time. Pre-training models is Pubmed Bert.
Subtask3-CR-EN-1 (ADE): Use multi-class classification model with two stage models (5 optimal models) and full data trained model ensemble.
Subtask3-CR-EN-2 (ADE): Use multi-class classification model with two stage models (6 optimal models) and full data trained model ensemble.
Subtask3-CR-EN-3 (ADE): Use multi-class classification model with only -fold cross-validation models and and full data trained model ensemble.
Subtask3-CR-EN-4 (ADE): Use multi-class classification model with two stage models (6 optimal models) and full data trained model ensemble (different weights).
Subtask3-CR-EN-5 (ADE): Use prompt learning model with only -fold cross-validation models and and full data trained model ensemble.
Subtask3-CR-EN-6 (ADE): Use prompt learning model with only -fold cross-validation models and and full data trained model ensemble and some manual summary rules.
The performance of our model with different technical points on the Subtask1-RR-EN test set for subtask3 are listed in Table 10.

6 CONCLUSIONS

In this paper, For Few-source NER, we propose a approaches which rely on span-based method and prompt learning. We compared previous model structures on different pre-training models. We conclude that selecting the correct pre-training model and a model structure more suitable for the task can achieve relatively good performance. Data augmentation is generally more obvious in the case of less training data. In addition, the combination results of different methods are not necessarily positive, so in the final combination, it is necessary to fully combine and fully verify the influence of different combinations, so as to obtain the optimal results.

For ADE, we mainly adopt two methods: multi-class classification and prompt learning. For multi-class classification, we try a two-stage training strategy to solve the long tail distribution problem. Note that the performance of the model in adverse drug reaction detection is greatly improved by creating related tasks. For prompt learning, we design the patterns and verbalizer for model. In addition, We also explore some data augmentation strategies in this task. However, the test results is not good in tail classes. We will be committed to solving the long tail distribution problem.

REFERENCES

[1] Shuntaro Yada, Yuta Nakamura, Shoko Wakamiya, and Eiji Aramaki. Real-mednlp: Overview of real document-based medical natural language processing task. In *Proceedings of the 16th NTCIR Conference*, 2022.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[3] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[4] Liang Xu, Qianqian Dong, Yixuan Liao, Cong Yu, Yin Tian, Weitang Liu, Lu Li, Caiquan Liu, Xuanwei Zhang, et al. Cluener2020: fine-grained named entity recognition dataset and benchmark for chinese. *arXiv preprint arXiv:2001.04351*, 2020.

[5] Markus Eberts and Adrian Ulges. Span-based joint entity and relation extraction with transformer pre-training. *arXiv preprint arXiv:1909.07755*, 2019.

[6] Zexuan Zhong and Danqi Chen. A frustratingly easy approach for entity and relation extraction. *arXiv preprint arXiv:2010.12812*, 2020.

[7] Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. Template-based named entity recognition using bart. *arXiv preprint arXiv:2106.01760*, 2021.

[8] Shuntaro Yada, Ayami Joh, Ribeka Tanaka, Fei Cheng, Eiji Aramaki, and Sadao Kurohashi. Towards a versatile medical-annotation guideline feasible without heavy medical knowledge: Starting from critical lung diseases. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4565–4572, Marseille, France, May 2020. European Language Resources Association.

[9] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition, 2020.

[10] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018.

[11] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples, 2019.

[12] Timo Schick and Hinrich Schütze. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*, 2020.

[13] Emily Alsentzer, John R. Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew B. A. McDermott. Publicly available clinical bert embeddings, 2019.

[14] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23, 2021.

[15] Chen Lin, Timothy Miller, Dmitriy Dligach, Steven Bethard, and Guergana Savova. EntityBERT: Entity-centric masking strategy for model pretraining for the clinical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 191–201, Online, June 2021. Association for Computational Linguistics.

[16] Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. Prompt-learning for fine-grained entity typing. *arXiv preprint arXiv:2108.10604*, 2021.

[17] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution. *arXiv preprint arXiv:1707.07045*, 2017.

[18] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition, 2020.

[19] Tong Wang, Yousong Zhu, Chaoyang Zhao, Wei Zeng, Jinqiao Wang, and Ming Tang. Adaptive class suppression loss for long-tail object detection. *CoRR*, abs/2104.00885, 2021.