Shorten the Long Tail for Rare Entity and Event Extraction

Pengfei Yu, Heng Ji
University of Illinois Urbana-Champaign {pengfei4,hengji}@illinois.edu

Abstract

The distribution of knowledge elements such as entity types and event types is long-tailed in natural language. Hence information extraction datasets naturally conform a long-tailed distribution. Although imbalanced datasets can teach the model about the useful real-world bias, deep learning models may learn features not generalizable to rare or unseen mentions of entities or events during evaluation, especially for rare types without sufficient training instances. Existing approaches for the long-tailed learning problem seek to manipulate the training data by re-balancing, augmentation or introducing extra prior knowledge. In comparison, we propose to handle the generalization challenge by making the evaluation instances closer to the frequent training cases. We design a new transformation module that transforms infrequent candidate mention representation during evaluation with the average mention representation in the training dataset. Experimental results on classic benchmarks on three entity or event extraction datasets demonstrate the effectiveness of our framework. [1]

1 Introduction

Long-tailed distributions are common in natural language processing tasks. This natural phenomenon of long-tailed distribution is formulated as Zipf's Law (Reed, 2001). For information extraction, knowledge elements such as entities, relations and events typically conform a long-tailed distribution in natural language. This leads to the imbalanced distribution of types in the benchmark datasets unless manual manipulation is performed to balance the dataset. We show the entity type distribution of an entity extraction dataset (Few-NERD (Ding et al., 2021)), and the event type distribution of two event extraction datasets (ACE 2005 (Walker et al., 2006) and MAVEN (Wang et al., 2020)) in Figure 1. All three datasets have similar long-tailed distributions. Among them, MAVEN and Few-NERD are two relatively large-scale datasets, but the training mentions are still concentrated on a small number of frequent types and majority of the entity/event types are rare types without sufficient training examples.

Long-tailed distribution is a generic problem not limited to NLP research. In fact, a lot of previous work on the long-tailed learning is in the computer vision domain (Lin et al., 2017; Kang et al., 2020). These approaches consider the imbalance in the type distribution as the major problem and are mostly based on balancing the datasets by upweighting the rare types and downweighting the frequent types. However, we argue that this line of research faces significant challenges when applying to the information extraction task. First, balancing the dataset breaks the real-world long-tailed prior in the datasets and may lead the model to mistakenly predict the long-tailed types too often. We observe this phenomenon in our experiments in Section 3 especially for the Classifier Re-training baseline (Kang et al., 2020). Second, due to the lack of effective data augmentation approaches as in the computer vision domain, upweighting the training mentions will only make the model to repetitively learn the same set of instances and aggravate the overfitting problem.

Instead of the imbalance in distribution, the real challenge of long-tailed learning for NLP is knowledge insufficiency, i.e. long-tailed datasets don't contain enough examples to acquire generalizable knowledge for the rare types. Learned models tend to capture features such as event trigger words for rare types and fail to generalize well to rare or even unseen cases (unseen trigger words) of those types during evaluation. For instance, if an event extraction model is trained on a dataset where all `Acquit` events are triggered by the keyword *acquit*, it may not identify the `Acquit` event trig-

[1] Code is available at `https://github.com/Perfec-Yu/LongTailIE`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1339–1350
May 2-6, 2023 ©2023 Association for Computational Linguistics

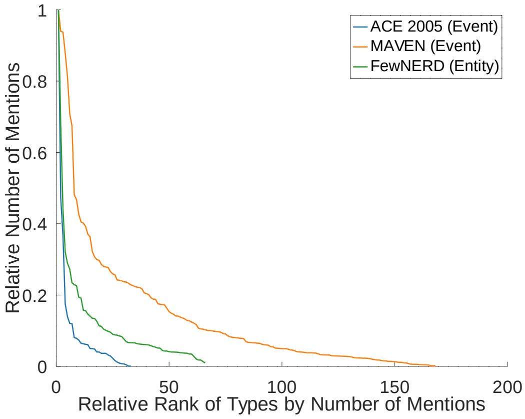


Figure 1: Distribution of types in ACE 2005 event dataset, MAVEN event dataset and the Few-NERD entity dataset. Y-axis is the number of training mentions divided by that of the most frequent type. X-axis is the rank of types by number of mentions.

gered by **walk free** in *The adjudicator allowed the criminal to **walk free***. Under this perspective, existing work tackling the long-tailed problem in information extraction (Han et al., 2018; Zhang et al., 2019) incorporates external structural knowledge to help learning rare types. However, the selection of the structural knowledge requires expertise in the target ontology and is not easily transferable to other ontologies. Another more recent line of work (Tang et al., 2020; Nan et al., 2021) considers the causal inference (Pearl et al., 2000) approach to solve the long-tailed problem. They aim to avoid the learning of spurious correlations between the input features and the labels in the limited training data for rare types. This is usually achieved by removing the effect of a manually selected confounding factor in prediction through deconfounded learning methods such as backdoor adjustment and inference based on total direct effect (Pearl et al., 2000). Although the model is guided to avoid spurious correlation through causal-inference-based approaches, learning the generalizable features with real causalities is still challenging due to the knowledge insufficiency problem.

In this work instead of confronting the knowledge insufficiency problem directly, we propose to bypass the problem by transforming the evaluation instances that require additional knowledge into more familiar instances that are closer to the frequent training instances. We decompose the input sentence $\{t_0, t_1, \ldots, t_n\}$ for the event/entity prediction task into two parts: a candidate token $e = t_i$ for the i-th token and surrounding contextual tokens $c = \{t_j\}_{j=0}^{i-1} \bigcup \{t_j\}_{j=i+1}^{n}$. Instead of making predictions for an event/entity type y_k solely based on $P(y_k|e, c)$, we propose to also consider predictions on averaged training inputs $P(y_k|c, \boldsymbol{r}_k)$, where $\boldsymbol{r}_k$ is the average training hidden representation of the type y_k. Our framework combines these two predictions using a transformation module. The transformation module computes weights g_e for the original candidate token e and g_k for each event type representation r_k. Moreover, instead of simply combining probabilities as $g_e P(y_k|e, c) + g_k P(y_k|c, \boldsymbol{r}_k)$, we use two weights g_e and g_k to combine in the representation layer for $\boldsymbol{e}$ and $\boldsymbol{r}_k$ to produce $P(y_k|g_e\boldsymbol{e} + g_k\boldsymbol{r}_k, c)$. The transformation module decides the weights g_e and g_k based on the frequency of the candidate mention in the training dataset and the cosine similarity between the candidate mention representation and all event types' average training mention representations in the ontology. Experimental results on three benchmark datasets demonstrate the effectiveness of our framework. Additionally, we found our approach, though derived from a different motivation, can be interpreted as a backdoor adjustment approach (Pearl et al., 2000) as shown in Section 2.5, which gives another interpretation that our approach improves long-tail learning by facilitating the model to learn the true correlation between the candidate text span and the event/entity type. We also empirically found that our approach facilitates the model to capture more generalizable features, which aligns with the goal of causal inference approaches.

To summarize, we propose a new approach for learning from the long-tailed datasets for entity extraction and event extraction. Our contributions are two-fold:

- We provide a new framework by bypassing the knowledge insufficiency problem in long-tailed learning and propose a novel learning framework in this perspective.
- We found that our framework aligns theoretically with the causal inference approach and can facilitate the model to capture more generalizable features.

2 Approach

2.1 Task Definition

In this work we take the task of both entity extraction and event extraction as sequence labeling

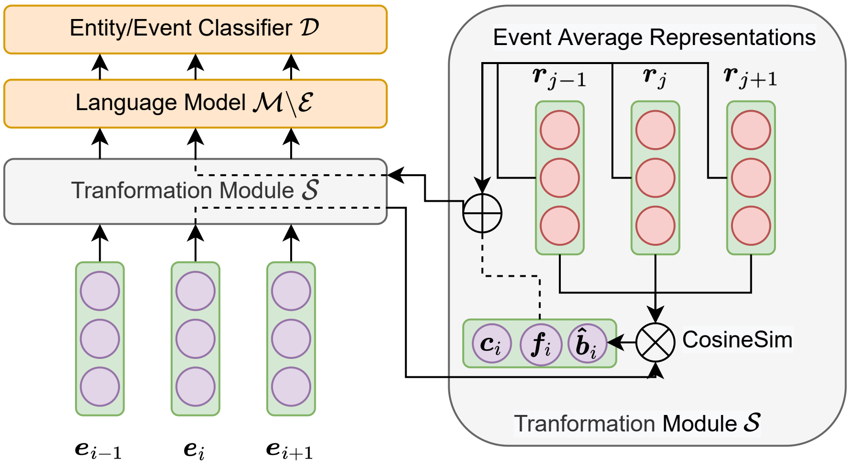


Figure 2: Overall long-tail learning information extraction framework. We ignore the original token t_i and only show the token embeddings $\boldsymbol{e}_i$ in this figure for conciseness. Here we use $\mathcal{M}\backslash\mathcal{E}$ to indicate the language model layers after the embedding layer and $\mathcal{D}$ as the classifier heads for the sequence labeling task. The transformation module $\mathcal{S}$ combines rare candidate mentions with average training mentions. The inputs to the transformation module includes the frequency feature $\boldsymbol{f}_i$, the similarity features $\hat{\boldsymbol{b}}_i$ between the token embedding $\boldsymbol{e}_i$ and the average training embeddings $\{\boldsymbol{r}_j\}_{j=1}^K$, and the context features $\boldsymbol{c}_i$.

problems. The input to the model $\mathcal{H}$ is a sequence of tokens $x = \{t_0, t_1, \ldots, t_n\}$, and the model predicts a label for each token $\mathcal{H}(x) = \{l_0, l_1, \ldots, l_n\}$. Each l_i is either one of the entity/event types or the `None` type for none-entity or none-event tokens.[2]

2.2 Overview of the Learning Framework

The learning of our proposed long-tailed extraction framework includes two steps. First, we finetune a language model with an additional classifier head for the sequence labeling problem without special treatments for the long-tail problem. In the second step, we train the transformation module while fixing the parameters of the language model. We adopt this two-stage training approach because the transformation module requires the language model's representations as inputs as we will show in Section 2.4. The finetuning step should render these representations as more task-specific features, which we redeem to be helpful for the learning of the transformation module. We will introduce these two steps in the following sections. The overall architecture is shown in Figure 2 and the two-staged training strategy is illustrated in Figure 3.

2.3 Finetuning Step

Given an input sequence $x = \{t_0, t_1, \ldots, t_n\}$, we first encode it with a pretrained language model $\mathcal{M}$ into $\mathcal{M}(x) = \{\boldsymbol{t}_0, \boldsymbol{t}_1, \ldots, \boldsymbol{t}_n\}$. We then adopt a linear discriminator $\mathcal{D}$ to predict the label for each token

$$l_i = \arg\max \mathcal{D}(\boldsymbol{t}_i). \tag{1}$$

We finetune $\mathcal{M}$ and $\mathcal{D}$ with the cross entropy loss.

2.4 Learning the Transformation Module

The transformation module $\mathcal{S}$ transforms the representation for each token t_i into a weighted combination of its original representation and the average training representation of each event type. Our goal is to compute weights g_e, g_k for the final prediction $P(y_k | g_e\boldsymbol{e} + g_k\boldsymbol{r}_k, c)$ for each type y_k. However, this would require feeding the inputs multiple times to the language model, which is inefficient. Hence, we simplify the prediction to $P(y_k | g_e\boldsymbol{e} + \sum_{i=1}^k g_k\boldsymbol{r}_k, c)$. In other words, our transformation module combines the average representations of all types together in the hidden layer. We can essentially perform the transformation in any hidden layers of the pretrained language model $\mathcal{M}$. In this work, we perform transformation in the embedding layer to take full advantage of $\mathcal{M}$'s capability of encoding contextual information.

Specifically, let $\mathcal{E}(t_i) = \boldsymbol{e}_i$ be the embedding representation of the token t_i. Let $\{\boldsymbol{r}_j\}_{j=1}^K$ be the average training embedding representation for the entity/event type j, i.e.,

$$\boldsymbol{r}_j = \frac{1}{\#j} \sum_{\{t | l(t) = j\}} \mathcal{E}(t). \tag{2}$$

[2] We adopt the IO labeling schema instead of BIO labeling schema to be consistent with the annotations in the Few-NERD dataset (Ding et al., 2021)

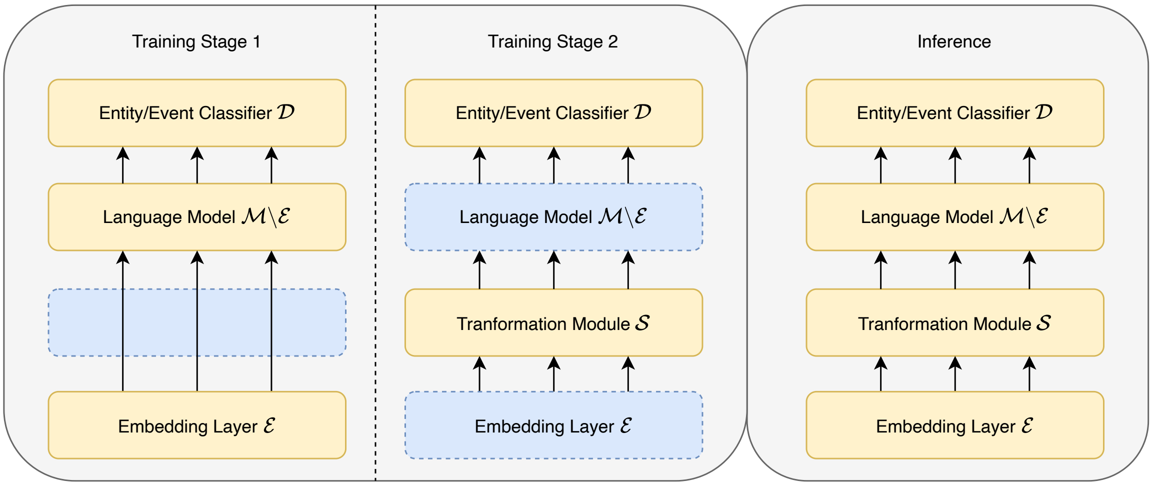

Training Stage 1
Entity/Event Classifier $\mathcal{D}$
Language Model $\mathcal{M}\backslash\mathcal{E}$
Embedding Layer $\mathcal{E}$
Training Stage 2
Entity/Event Classifier $\mathcal{D}$
Language Model $\mathcal{M}\backslash\mathcal{E}$
Tranformation Module $\mathcal{S}$
Embedding Layer $\mathcal{E}$
Inference
Entity/Event Classifier $\mathcal{D}$
Language Model $\mathcal{M}\backslash\mathcal{E}$
Tranformation Module $\mathcal{S}$
Embedding Layer $\mathcal{E}$

Figure 3: Illustration of the training and inference stages. There are two stages of the training on the left. Modules with dashed lines and blue background are not updated during the corresponding training stage.

Here $\{t|l(t) = j\}$ refers to the training tokens with the label j and $\#j$ is the total number of such tokens. The transformation module computes weights to combine each $\boldsymbol{e}_i$ with $\{\boldsymbol{r}_j\}_{j=1}^K$. We use the following information as input to compute the weights for combination

- t_i's frequencies of being labeled as entity/event types : $\boldsymbol{f}_i = [\exp(\gamma f_{ij})]_{j=1}^K$. Here we rescale the original frequencies inspired by (Lin et al., 2020). We use $\gamma = 0.1$ following (Lin et al., 2020). We expect the tranformation module to provide most help in rare cases when $\boldsymbol{f}_i$ is small.

- The similarity of t_i's embedding $\boldsymbol{e}_i$ with average training embeddings $\boldsymbol{r}_j$: $\boldsymbol{b}_i = [\cos(\boldsymbol{e}_i, \boldsymbol{r}_j)]_{j=1}^K$. If $\boldsymbol{b}_i$ contains large values, t_i is close to some event type's average embedding and potentially not a rare mention. On the other hand, if $\boldsymbol{b}_i$ is very small, t_i deviates significantly from some event type's average embedding and the transformation may significantly alter the meaning of the input. Taking these into consideration, we propose to model the similarity value $\boldsymbol{b}_i$ with a quadratic function, i.e. we use $\hat{\boldsymbol{b}}_i = [\boldsymbol{b}_i; \boldsymbol{b}_i^2]$ as the similarity features to the transformation module.

- The context representation $\boldsymbol{c}_i$, which is computed by the pretrained language model $\mathcal{M}$ while masking t_i in the input sequence x. Hence $\boldsymbol{c}_i$ is independent of the token t_i. This is to make sure that the transformation module is not overfitted only on the seen candidate tokens during training, because our goal is to apply the transformation module to rare cases in the evaluation corpus.

We decompose the computation of these weights into two steps. We first compute aggregation weights $\{\alpha_j\}_{j=1}^K$ for $\{\boldsymbol{r}_j\}_{j=1}^K$ with an attention module based on the context $\boldsymbol{c}_i$:

$$\begin{aligned} \alpha_j &= \frac{\exp(\langle \boldsymbol{a}_j, \boldsymbol{c}_i \rangle)}{\sum_{k=1}^K \exp(\langle \boldsymbol{a}_k, \boldsymbol{c}_i \rangle)}, \\ \boldsymbol{s}_i &= \sum_j \alpha_j \boldsymbol{r}_j \end{aligned} \quad (3)$$

Here $\{\boldsymbol{a}_j\}_{j=1}^K$ are trainable weights. We then feed all three features $[\boldsymbol{f}_i; \hat{\boldsymbol{b}}_i; \boldsymbol{c}_i]$ into a linear layer with the sigmoid activation to compute two gating scores $(g_e, g_s) \in [0,1]^2$. The final embedding representation is computed as

$$\boldsymbol{h}_i = g_e \boldsymbol{e}_i + g_s \boldsymbol{s}_i. \quad (4)$$

We then substitute $\boldsymbol{e}_i$ with $\boldsymbol{h}_i$ for the following layers in $\mathcal{M}\backslash\mathcal{E}$.[3]

We train the transformation module $\mathcal{S}$ using the same cross entropy loss as the finetuning step. We also compute the cross entropy on the attention weights of the entity/event types in Equation (3) as an additional loss. We fix the parameters of the finetuned language model $\mathcal{M}$. This ensures that the similarity features $\hat{\boldsymbol{b}}_i$ and the context representations $\boldsymbol{c}_i$ remain fixed during the learning of $\mathcal{S}$. We provide training details in the Appendix.

[3] From Equation (4) we essentially have $g_k = g_s \alpha_k$

2.5 Connection to Causal Inference

We found our architecture can be mathematically intepreted as a backdoor adjustment (Pearl et al., 2000) method in causal inference, which is used to mitigate the effect of some confounding factor $\mathcal{U}$ in making decisions. In our case, it can be formulated as the following prediction probability

$$P(l_i|do(t_i)) = \sum_u P(l_i|t_i, \mathcal{U}=u)P(\mathcal{U}=u), \tag{5}$$

where u is the value space of $\mathcal{U}$. We refer readers to (Pearl et al., 2000) for the derivation of the above equation[4]. If we select the confounding factor $\mathcal{U}$ as the type prediction based on the context around the token t_i (independent of t_i), Equation (5) becomes

$$P(l_i|do(t_i)) = \sum_{j=1}^{K} P(l_i|t_i, j)P(j|\boldsymbol{c}_i), . \tag{6}$$

The attention weights $\{\alpha_j\}_{j=1}^{k}$ in Equation (3) is can be considered as probabilities of event types dependent on $\boldsymbol{c}_i$, thus can be seen as $P(j|\boldsymbol{c}_i)$ in the above equation. If we further model $P(l_i|t_i, j) = P(l_i|\boldsymbol{e}_i + \boldsymbol{r}_j)$ and apply the Normalized Weighted Geometric Mean (NWGM) approximation following (Yue et al., 2020),

$$P(l_i|do(t_i)) \approx P(l_i|\boldsymbol{e}_i + \sum_{j=1}^{k} \alpha_j \boldsymbol{r}_j). \tag{7}$$

In our framework, we are essentially computing

$$P(l_i|t_i) = P(l_i|g_e\boldsymbol{e}_i + g_s \sum_{j=1}^{k} \alpha_j \boldsymbol{r}_j). \tag{8}$$

This means that α_j composes the deconfounding priors $P(\mathcal{U})$ and (g_e, g_s) serve as the switch for the backdoor adjustment decisions. We expect the model to make decisions on rare or unseen mentions with the backdoor adjustment, and rely on the model's own prediction with more frequent cases.

3 Experiments

3.1 Datasets and Evaluation

We experiment on an entity extraction dataset, Few-NERD (Ding et al., 2021) and two event extraction datasets, ACE 2005 (Walker et al., 2006) and MAVEN (Wang et al., 2020). We provide data statistics in Table 1. Few-NERD, MAVEN and ACE include 67, 168, and 33 types respectively. For ACE 2005, we split the dataset such that all event types are covered in the evaluation data which is different from the splits used in previous work. For Few-NERD, we used the "supervised" split for experiments.

For evaluation, we use the macro F1 score as the main metric to reflect the influence of long-tail types. We also report micro F1 for additional reference. We provide macro recall, macro precision, micro recall and micro precision in Appendix. Previous work also reports scores on subsets of rare types (Nan et al., 2021). Instead of manually determine a frequency threshold for the "rarest" types, we plot curves of F1 scores on all types in Figure 4. All results are averaged over three runs using different random seeds.

3.2 Methods in Comparison

We group previous work on long-tailed learning into two categories: balancing-based approaches and causal inference approaches. For balancing-based approaches, we mainly follow (Kang et al., 2020) and implemented Classifier Re-training (CRT), Nearest Class Mean classifier (NCM), τ-normalized classifier (τ-norm) and Learnable weight scaling (LWS). We also implemented the focal loss approach (Lin et al., 2017). For causal inference work, we implemented Momentum (Tang et al., 2020) and CFIE (Nan et al., 2021). Although (Nan et al., 2021) also experimented on the ACE 2005 and MAVEN, we re-implemented[5] their methods due to the difference in splits and evaluation strategies. CFIE also has different designs for the entity extraction and event extraction models. We didn't re-implement the entity model due to insufficient details. We also compare with a **Vanilla** baseline which is the sequence labeling model without the transformation module.

3.3 Main Results

We show our main results in Table 2 and Figure 4. From Table 2, our framework achieves the best macro F1 scores across three datasets. We also noticed that our approach improves macro F1 scores without suffering from inferior micro F1 scores. Al-

[4] Note that the common formulation for $P(l_i|t_i)$ would be substituting $P(\mathcal{U}=u)$ with $P(\mathcal{U}=u|t_i)$

[5] Due to insufficient details in their paper and released code, we implemented their framework to the best of our knowledge. CFIE also requires named entity extraction results as part of the inputs to event extraction. Therefore it is not clear how the entity extraction model is designed.

Dataset	Few-NERD		MAVEN		ACE 2005	
	#Sentences	# Mentions	#Sentences	# Mentions	#Sentences	# Mentions
train	131,767	340,387	32, 431	77,993	16,807	4,254
dev	18,824	48,770	8,042	18,904	2056	500
test	37,648	96,902	9,400	21,835	1,930	570

Table 1: Dataset statistics.

though the Momentum baseline performs closely to our approach in ACE Macro F1. However, Momentum has a much worse micro F1 sacrificing frequent types and worse performance on the other two datasets. These results also show the superiority of our approach and indicate that our model's performance on frequent types is not harmed. This is because we do not tackle the long-tailed learning problem in terms of balancing that may affect the natural distribution in the dataset. Instead, we transform the long-tailed evaluation samples into frequent training samples with the transformation module.

Although macro F1 metric puts extra stress on the long-tailed types, the performance on the rare types is still dilated by frequent types especially for datasets with a large number of types such as MAVEN. In order to further investigate the performance on different subsets of event types, we show in Figure 4 macro F1 improvements (over the `Vanilla` baseline) for the top X fraction of the event types with the fewest training examples. We notice that our approach, together with other approaches, indeed improves the learning with long-tailed types when X is small. We notice that our approach can significantly outperform the baseline for the less frequent types with a maximum of over 10% on ACE 2005 event dataset. We found our approach has the best performance of the moderately long-tailed event types compared with other long-tailed learning approaches. For extremely long-tailed types, our performance is also close to the best method `Momentum` (Tang et al., 2020). One possible reason of our model's ineffectiveness on extremely long-tailed types is that we cannot learn reliable representation $\boldsymbol{e}_i$ to compute similarity values $\hat{\boldsymbol{b}}_i$ for the transformation module. Another possible reason is that the extreme long-tailed types don't have enough training instances to provide an informative average training representation r_k. A potential solution would be introducing external knowledge (e.g., a few keywords as candidate event triggers or entity mentions) to enrich the r_k. Moreover, our approach is more consistent than other methods when X grows larger. This further validates that our approach improves the long-tailed types without sacrificing the performance of frequent event types. For more references, we also provide top 10 event types in MAVEN dataset where our method achieves most F1 performance gain on in Appendix. Majority of those types are rare types.

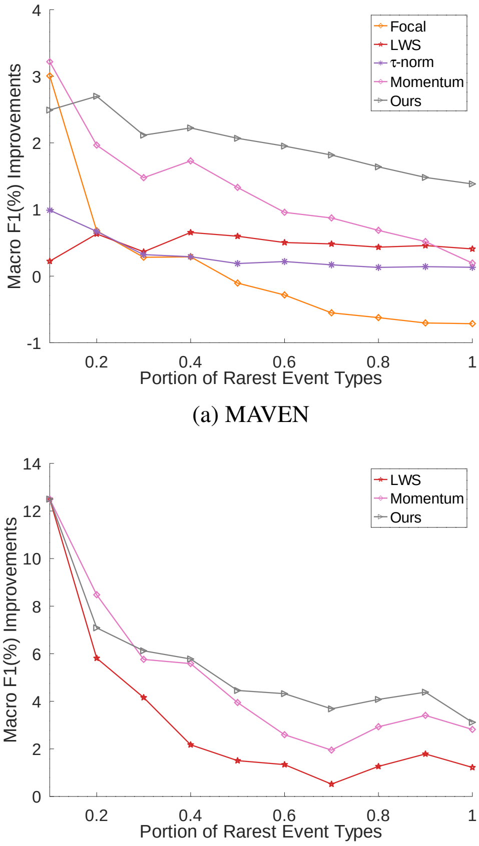
3.4 Improving Classifier Features

In our second training stage, we finetune the entity/event classifier head together with the transformation module. Finetuned classifiers from our approach should benefit from the connections of our approach and the backdoor adjustment approach for causal inference in Section 2.5, which encourages the model to capture generalizable features instead of surface correlations. To test the finetuned classifiers alone, we evaluate our framework with the transformation module disabled, i.e. forcing $(g_e, g_s) = (1, 0)$ in Equation (4). The forward architecture becomes exactly the same as the `Vanilla` model. In Table 3, we observe that finetuned classifier alone outperforms the `Vanilla` baseline. Since we fixed the language model in the second stage, the improvements purely come from the finetuned classifiers that have learned to avoid surface features for prediction.

In addition to the interpretation of the improvements from the causal inference theory, we give a more intuitive explanation based on Equation (4). During the second training step, Equation (4) can be seen as changing the original candidate representation with an interpolation of itself and other candidates' representations of the same event type. This is similar to augmenting the dataset by interchanging mention spans of the same type across sentences, though we interchange it with the average representation of that type in the entire training dataset. The classifier may benefit from this "aug-

Dataset	Few-NERD		MAVEN		ACE 2005	
	Macro F1	Micro F1	Macro F1	Micro F1	Macro F1	Micro F1
Vanilla	62.4	**67.8**	60.1	67.1	60.4	72.7
Focal	62.5	**67.8**	59.4	65.3	59.9	**73.9**
CRT	44.9	54.3	51.2	54.3	60.2	71.8
LWS	61.4	66.9	60.5	66.7	60.9	72.5
τ-norm	62.4	**67.8**	60.2	67.1	60.2	72.7
NCM	51.5	50.4	57.5	61.2	57.8	72.2
CFIE	-	-	56.3	61.2	50.6	63.2
Momentum	59.7	63.9	60.1	66.4	62.1	68.9
Ours	**62.6**	**67.8**	**61.2**	**67.4**	**62.4**	72.9

Table 2: Macro and micro F1 scores (in %) on three datasets. References to the methods above is provided in Section 3.2.



(a) MAVEN

(b) ACE 2005

Figure 4: Performance difference compared with Vanilla w.r.t the portion of rarest event types. We first rank event types by the number of training mentions from low to high, and then compute average F1 scores (minus that of the Vanilla baseline) for the first X fraction of the event types. We omitted those approaches that are significantly lower than Vanilla.

mentation” to capture more general features.

Method	Ours	No Sub	Vanilla
Macro F1	62.4	61.9	60.4

Table 3: Performance (in %) with the transformation module disabled (No Sub) on the ACE 2005 dataset.

3.5 Case Study

We also show some examples that the Vanilla baseline misses but our approach correctly identifies and classifies the candidate mentions in Table 4. We also visualize the transformation weights (g_e, g_s) in Equation (4). As discussed in Section 3.3, our model is the best at handling moderately long-tailed types. For these two examples, the event Start-Position in the first case has 94 mentions and became appears only 2 times as the trigger. In the second case, the Action appears 709 times, but only 16 of them are triggered by undertaken[6]. It is worth mentioning that although g_s is not as large as g_e in both cases, we found that they are indispensable since the model fails on both cases if we disable the transformation module by forcing $g_s = 0$.

4 Related Work

4.1 Balancing-based Long-tailed Learning for Computer Vision

Long-tailed learning is closely related to imbalanced learning. In the computer vision domain, Lin et al. (2017) proposes the focal loss to handle

[6] As a comparison, war appears 270 times as the trigger of Attack in ACE 2005 and storm appears 757 times as the trigger of Catastrophe in MAVEN.

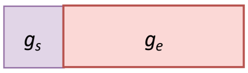
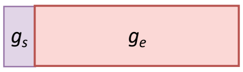
Input Sentence	Dataset	Transformation Weights
This upcoming visit to Russia will be my first trip aboard since I **became** president of China. (Event: `Start-Position`)	ACE	
The operation was **undertaken** so that Allies could secure a beachhead. (Event: `Action`)	MAVEN	

Table 4: Examples of missing event triggers by the `Vanilla` model. Transformation weights (g_e, g_s) in Equation (4) are visualized.

the imbalanced learning problem in object detection. The focal loss downweights the loss terms for confident training samples so that the model predicts high probabilities for the gold standard labels. In addition to this, some work (Zhang et al., 2019; Rebuffi et al., 2017), though not focusing on the long-tailed learning problem, adopts special training strategies to tackle the imbalance in the datasets. Kang et al. (2020) summarize these approaches and evaluate them with the computer vision tasks. These approaches are mostly based on balancing the datasets by upweighting the rare types and downweighting the frequent types, either by modifying the sampling strategy or associating weights to the loss terms. These methods first learn the feature extractor with the original imbalanced distribution and then perform special treatments to the classification layer. Classifier Re-training (CRT) retrains the classification layer by sampling examples from each type uniformly. Nearest Class Mean classifier (NCM) uses the average training features as the type-level features and uses certain distance metrics to perform the nearest neighbor classifcation. τ-normalized classifier (τ-norm) normalizes the type weights in the classification layer. This can make sure the weights for frequent types are not significantly larger than rare types. Learnable weight scaling (LWS) is similar to τ-norm but learns the normalization weights while sampling examples from each type uniformly.

4.2 Long-tailed Learning with External Knowledge for IE

Balancing-based methods are usually ineffective in information extraction as shown in our experiments. Existing work usually tackles the long-tailed problem in information extraction (Han et al., 2018; Zhang et al., 2019) by incorporating external structural knowledge. Han et al. (2018) adopt hierarchical structures among relation types to transfer knowledge from the frequent relation types to their siblings. Zhang et al. (2019) incorporate label semantics and knowledge graph embeddings to transfer knowledge from frequent types to rare types. Required expertise in selecting external knowledge limits the generalization of these methods. (Yu et al., 2021) also leverages correlation between event types to help the learning of rare event types in the context of lifelong learning.

4.3 Causal Inference and Its Application to Long-tailed Learning

Due to the potential of causal inference (Pearl et al., 2000) theory to reduce the spurious correlation, there have been explorations on its application in machine learning (Lopez-Paz et al., 2017; Magliacane et al., 2018; de Haan et al., 2019; Bengio et al., 2020; Yang et al., 2020; Li et al., 2021; Park et al., 2021). Since the the spurious correlation is more common in limited data, some work attempts to interpret the long-tailed learning problem under a causal inference framework. The core component of this interpretation is to find a confounding factor that affects the distribution of the input features and output labels at the same time. Tang et al. (2020) consider the training momentum of the gradients to be the confounding factor and proposed the corresponding deconfounded training with the backdoor adjustment approach. They use the total direct effect (TDE) for inference, which essentially lowers the probabilities of frequent types systematically. Nan et al. (2021) work on the information extraction tasks and take a set of linguistic features as the confounding factor. They also adopt an inference approach similar to TDE to lower the probabilities of frequent types.

4.4 Related Few-shot Learning Methods

Few-shot learning aims at training models with a small number of instances, which is similar to the goal of improving rare types in long-tailed learning. Some few-shot learning methods also overlap with long-tailed learning methods. Snell et al. (2017) proposed a prototypical network that has a similar framework as NCM for long-tailed learning. Yue et al. (2020) proposed a backdoor adjustment approach based on the causal inference thoery to reduce the spurious correlation in the model, since the few-shot learning also has the limited data size problem. However, few-shot learning usually works on the N-way K-shot setting. This makes these approaches usually not directly applicable to the long-tailed learning scenario.

5 Conclusions and Future Work

In this work we propose a new long-tailed learning framework for entity and event extraction by candidate transformation. We design a novel transformation module to convert representations of rare or unseen mentions during evaluation into representations of average training mentions. Experimental results have validated the effectiveness of our framework. Our framework can significantly improve the performance on long-tailed types, and outperform other long-tailed learning methods especially for moderately long-tailed types. Moreover, our framework does not sacrifice the performance on frequent types. We also discover the connections between our learning framework and the backdoor adjustment in the causal inference theory We empirically observe that our training strategy can improve the model's capability in capturing more generalizable features, which aligns with the causal inference theory. In the future, we will explore: (1) adapt the concept of transformation module to other NLP tasks; (2) based on our connection with the causal inference theory, it is possible to design a better transformation module by choosing a better confounding factor other than the context information.

6 Limitations

In terms of the framework design, our current design of the framework is only applicable to the sequence labeling task, although we believe it can be adapted to other NLP tasks without significant modifications. Besides, our framework should be most helpful if the semantics of the elements in the sequence of the same type are close to each other and thus may require additional modifications in more heterogeneous cases, such as vision-language models where visual features may have the same type as the textual token embeddings.

In terms of time efficiency, our framework will require an additional training stage to learn the transformation module, which will cost extra time. Since we fixed the language model, which composes majority of parameters in our framework, this additional cost is acceptable. We also recommend to pre-process input features (since they are fixed with respect to training samples during the second stage) to the transformation module to further reduce the time cost in the second stage.

7 Acknowledgement

This research is based upon work supported in part by U.S. DARPA LORELEI Program No.HR0011-15-C-0115, U.S. DARPA AIDA Program No. FA8750-18-2-0014 and KAIROS Program No. FA8750-19-2-1004. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Yoshua Bengio, Tristan Deleu, Nasim Rahaman, Nan Rosemary Ke, Sébastien Lachapelle, Olexa Bilaniuk, Anirudh Goyal, and Christopher J. Pal. 2020. A meta-transfer objective for learning to disentangle causal mechanisms. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Pim de Haan, Dinesh Jayaraman, and Sergey Levine. 2019. Causal confusion in imitation learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11693–11704.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-NERD: A few-shot named entity recognition dataset. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3198–3213, Online. Association for Computational Linguistics.

Xu Han, Pengfei Yu, Zhiyuan Liu, Maosong Sun, and Peng Li. 2018. Hierarchical relation extraction with coarse-to-fine grained attention. In *Proceedings of* *the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2236–2245, Brussels, Belgium. Association for Computational Linguistics.

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. 2020. Decoupling representation and classifier for long-tailed recognition. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Xin Li, Zhizheng Zhang, Guoqiang Wei, Cuiling Lan, Wenjun Zeng, Xin Jin, and Zhibo Chen. 2021. Confounder identification-free causal visual feature learning. *CoRR*, abs/2111.13420.

Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2999–3007. IEEE Computer Society.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

David Lopez-Paz, Robert Nishihara, Soumith Chintala, Bernhard Schölkopf, and Léon Bottou. 2017. Discovering causal signals in images. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 58–66. IEEE Computer Society.

Sara Magliacane, Thijs van Ommen, Tom Claassen, Stephan Bongers, Philip Versteeg, and Joris M. Mooij. 2018. Domain adaptation by using causal inference to predict invariant conditional distributions. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 10869–10879.

Guoshun Nan, Jiaqi Zeng, Rui Qiao, Zhijiang Guo, and Wei Lu. 2021. Uncovering main causalities for long-tailed information extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9683–9695, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jongjin Park, Younggyo Seo, Chang Liu, Li Zhao, Tao Qin, Jinwoo Shin, and Tie-Yan Liu. 2021. Object-aware regularization for addressing causal confusion in imitation learning. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 3029–3042.

Judea Pearl et al. 2000. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress*, 19:2.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5533–5542. IEEE Computer Society.

William J Reed. 2001. The pareto, zipf and other power laws. *Economics letters*, 74(1):15–19.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4077–4087.

Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. 2020. Long-tailed classification by keeping the good and removing the bad momentum causal effect. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. ACE 2005 multilingual training corpus LDC2006T06. Web Download. Philadelphia: Linguistic Data Consortium.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. MAVEN: A Massive General Domain Event Detection Dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1652–1671, Online. Association for Computational Linguistics.

Xu Yang, Hanwang Zhang, and Jianfei Cai. 2020. Deconfounded image captioning: A causal retrospect. *CoRR*, abs/2003.03923.

Pengfei Yu, Heng Ji, and Prem Natarajan. 2021. Lifelong event detection with knowledge transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5278–5290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhongqi Yue, Hanwang Zhang, Qianru Sun, and Xian-Sheng Hua. 2020. Interventional few-shot learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Guanying Wang, Xi Chen, Wei Zhang, and Huajun Chen. 2019. Long-tail relation extraction via knowledge graph embeddings and graph convolution networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3016–3025, Minneapolis, Minnesota. Association for Computational Linguistics.

A Appendix

A.1 Details on Two-stage Training

In the first training stage, all parameters in the language model $\mathcal{M}$and entity/event classifier head $\mathcal{D}$ are updated. In the second stage, we fix the entire language model and train both $\mathcal{D}$ and the transformation module $\mathcal{S}$. Apart from the cross entropy loss for entity/event extraction, we also apply loss to the weights $\{\alpha_j\}_{j=1}^{K}$ in Equation (3). This loss is only applicable to entity/event mention tokens, since $\{\alpha_j\}_{j=1}^{K}$ corresponds to entity/event types. For these mention tokens, we compute cross entropy between attention weights and the labeled types.

For the input features to the transformation module, we want to make the values of all three kinds of features to be in the similar range. Therefore, we adopt a batch normalization module to these features. We also add dropout with probability 0.2 to similarity features and frequency features, and dropout with probability 0.5 to the context features.

A.2 Implementation Details

For all experiments, we use learning rate to be $1e-5$ and batch size to be 8 to fit in a single Nvidia Tesla V100 GPU with 16GB memory. We evaluate performance after each epoch and select the best model based on the development performance. We use early-stop strategy with a patience of 5 epochs. We report average performance over 3 runs initialized with 3 different random seeds.

The approximate number of parameters is 3.5 million (RoBERTa). Added parameters from the transformation module is significantly less and dependant on the number of target entity/event types. Approximately the transformation module has 2,500 parameters. For the first stage training, it takes about 10-20 minutes to train an epoch on ACE 2005 data, 20-30 minutes on MAVEN data and about 40 minutes on Few-NERD. The time difference mainly comes from the total number of sentences in these datasets. However these are just rough estimations since the performance largely depends on the environmental factors such as temperatures and also the workload of other gpus/cpus in the same machine during training. In the second stage, we didn't preprocess any features in advance but block the backpropagation to language model parameters. In general it is 4-5 times faster than the first stage. We incorporate the transformation module by modifying the RoBERTa implementation from Transformers[7] Library.

A.3 Dataset Licenses

This dataset is licensed by LDC.[8] Membership is required for access. The dataset can be used for research purpose. Few-NERD dataset is distributed under the CC BY-SA 4.0 license. MAVEN dataset is ditributed under MIT License.

A.4 Additional Results

We include precision and recall scores in Table 5. Moreover, we include top 10 event types in the MAVEN dataset that our approach achieves most improvements over the Vanilla baseline, as well as their number of training mentions in Table.

[7] https://huggingface.co/docs/transformers/index

[8] https://www.ldc.upenn.edu

Dataset	Few-NERD		MAVEN		ACE 2005	
	Macro P,R	Micro P,R	Macro P,R	Micro P,R	Macro P,R	Micro P,R
Vanilla	60.1, 65.3	65.8, 70.0	61.9, 61.1	65.7, 68.6	63.6, 62.0	73.4, 72.1
Focal	60.6, 64.9	66.0, 69.8	57.8, 63.1	62.0, 69.1	61.9, 63.0	73.8, 74.1
CRT	40.7, 51.1	50.6, 58.8	41.1, 75.1	41.2, 79.8	58.6, 67.2	66.7, 77.9
LWS	58.1, 65.7	64.3, 69.7	60.1, 63.9	62.4, 71.8	62.5, 64.7	72.0. 73.0
τ-norm	60.8, 63.8	65.9, 69.9	62.2, 61.1	64.9, 69.4	63.5, 61.8	73.4, 72.0
NCM	56.0, 65.6	61.0, 43.3	58.9, 63.6	57.3, 65.7	64.2,56.5	76.1, 68.7
CFIE	-	-	50.9, 66.5	52.5, 73.5	59.7, 56.9	58.0, 69.7
Momentum	60.3, 65.4	59.7, 62.5	61.8, 61.9	66.3, 66.5	60.9, 68.3	66.9, 69.5
Ours	60.3, 65.4	65.8, 70.0	61.3,63.2	65.5, 69.4	63.5, 65.5	71.4, 74.6

Table 5: Macro and micro precision and recall scores (in %) on three datasets.

Event Type	# of Mentions	F1 (%) Imp.
Kidnapping	87	19.0
Body_movement	115	15.8
Emptying	124	15.1
Manufacturing	326	13.8
Scouring	32	13.2
Carry_goods	48	12.0
Military_operation	1,022	10.5
Practice	37	10.4
Labeling	35	10.0
Cure	71	8.4

Table 6: Top 10 event types in the MAVEN dataset that our approach achieves most improvements over the Vanilla baseline. We also include the number of training mentions and F1 score improvements (in %) in the second the third columns. MAVEN has a total of 168 event types. and our training split includes 77,987 training event mentions in total for all event types.