# Selecting the Right Experts: Generalizing Information Extraction for Unseen Scenarios via Task-Aware Expert Weighting

**Lubingzhi Guo[a,*], Javier Sanz-Cruzado[a] and Richard McCreadie[a]**

[a]University of Glasgow, Glasgow, United Kingdom
ORCID (Lubingzhi Guo): https://orcid.org/0009-0001-9283-5747, ORCID (Javier Sanz-Cruzado): https://orcid.org/0000-0002-7829-5174, ORCID (Richard McCreadie): https://orcid.org/0000-0002-2751-2087

**Abstract.** Information extraction (IE) systems aim to convert free text into structured knowledge that can be more easily and effectively used for a wide range of tasks, such as question answering or explanation generation. However, as text sources and information needs have diversified, developing domain-specific IE solutions is becoming less practical due to the lack of training data; resulting in a need for generalizable solutions that can perform IE over unseen data types and information needs. Current approaches augment an LLM with a single low-rank adapter (LoRA) via tuning over many tasks to attain some IE generalizability, but regularly fail when targeting information types not in the training data. We hypothesize that one of the reasons for this is IE-task-insensitivity, i.e. just telling the model what to look for in the prompt is insufficient context to guide the model. In this paper, we propose Task-Aware MoELoRA, a novel method that embeds an additional task signal into the IE process via a mixture-of-experts router. Through extensive experimentation over 35 IE datasets, we show that Task-Aware MoELoRA method significantly outperforms the LoRA baselines over the majority of unseen tasks, achieving gains of up to 8.2%.

## 1 Introduction

Information Extraction (IE) aims to identify and categorize structured information, such as different types of entities [52, 3], relations [53, 63] and events [46, 28], from unstructured natural language texts. The extraction of meaningful information from raw text predominantly follows the annotate-then-extract paradigm. In this approach, texts are manually or distantly labeled to train machine learning models, which are then used to automate extraction from new texts within the same domain. However, this approach typically leads to the development of specialized models tailored to specific datasets and domains, such as biomedicine [24], finance [17], or news [60]. To develop a model capable of universally solving different IE tasks, the Universal IE (UIE) framework [37] has been proposed to uniformly encode various extraction structures.

More recently, Large Language Models (LLMs) have been leveraged to facilitate more effective and uniform extraction across multiple tasks and domains, thereby promoting knowledge sharing and cross-domain adaptation [45, 30]. However, model performance generally falls short on unseen scenarios, with an average F1 score of 55% compared to 73% achieved in supervised settings [45]. This discrepancy may arise from the current practice in LLM-based UIE methods, which rely on a single shared low-rank adapter (LoRA [19]) to facilitate various downstream tasks. We hypothesize that using a single set of parameters for all types of input is insufficient, i.e. a single adapter is not expressive enough to dynamically emphasize the relevant knowledge learned across multiple datasets. We argue that adopting an architecture with multiple adapters allows for a more flexible configuration, which might be particularly beneficial when adapting to unseen datasets. Mixture-of-Experts (MoE) [22, 47, 8], integrated with multiple LoRA experts, is an architecture that capable of learning dynamic parameter sets for different tasks by leveraging a task-motivated gating mechanism [33, 5, 12, 36]. However, these studies perform gating using either predefined identifiers of datasets or their derived cluster embeddings, resulting in poor performance in low-resource scenarios where the model lacks any prior exposure to task-specific information. This represents an important limitation of existing gating functions – if emphasis is placed on the wrong experts via poor task disambiguation, then the probability of erroneous relations being produced significantly increases.

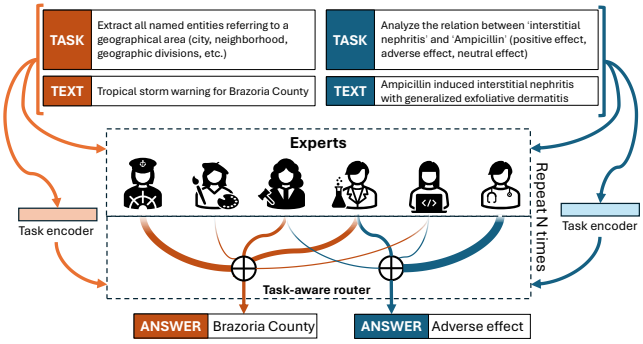


**Figure 1.** Task-Aware MoELoRA for UIE

To tackle the above challenge, we propose incorporating training dataset-agnostic semantic information from a general-purpose encoder into the router to guide the gating mechanism, thereby facilitating improved generalization across unseen IE tasks. This method enables better adaptability to semantically-related tasks and allows the model to transfer learned representations during training. Furthermore, by combining the task embeddings with token-level infor-

* Corresponding Author. Email: l.guo.1@research.gla.ac.uk

mation for routing, we introduce a task-aware routing mechanism into the dense MoELoRA architecture, enabling the model to learn multiple sets of parameters in a more fine-grained manner. In this paper, our contributions are threefold:

- We propose leveraging static semantic task embeddings to facilitate multi-task adaptation of LLMs with a Mixture of LoRA architecture. To the best of our knowledge, this is the first study to apply MoELoRA to the UIE task.
- We devise a Task-Aware MoELoRA architecture to improve the model generalization on unseen tasks, addressing the limitations of relying on a single LoRA adapter for IE tasks.
- Our proposed model outperforms the LoRA baseline on unseen datasets, achieving an 8.2% improvement in F1 score while maintaining comparable parameter size, with demonstrating consistent performance across 18 datasets.

## 2 Related Work

### 2.1 Universal Information Extraction (UIE)

The challenge of handling multiple information extraction tasks with a single model arises from the diversity of user-specified label schemas, which vary widely in structure, semantic granularity and domains. The UIE framework tackles this by approaching IE tasks as unified sequence-to-sequence problems, allowing models to generalize across schema variations. Lu et al. [37] first introduced a structured extraction language (SEL) representation to uniformly encode different IE tasks as a unified text-to-structure generation task. However, this method requires separate fine-tuning of the pre-trained model for each downstream task, which is not optimal for scenarios with limited annotations and restricts cross-task generalization. Recent studies have leveraged LLMs through instruction tuning to facilitate simultaneous learning across multiple tasks, thereby demonstrating promising generalization capabilities. In this configuration, IE tasks are represented as instruction-based generation tasks.

The model learns to follow unified task instructions and output examples, commonly formatted as natural language prompts [54] or code generation examples [27]. Sainz et al. [45] proposed a code schema-based approach that encodes various schemas as Python classes. They further integrated descriptive guidelines as comments, improving the ability of the LLM to follow annotation guidelines when generating structured information. Similarly, Li et al. [30] incorporated an additional code pre-training phase to improve the model's schema understanding ability. However, previous code schema-based UIE methods typically train a single adapter, which constrains the LLMs ability to learn diverse patterns from a wide range IE tasks and results in poor performance when targeting unseen labels. To address these limitations, we propose the Task-Aware MoELoRA architecture, which leverages explicit task guidelines to improve transferability from seen to unseen labels.

### 2.2 Mixture-of-LoRA Architecture

Low-Rank Adaptation (LoRA) [19], a Parameter-Efficient Fine-Tuning (PEFT) [41] method, reduces computational costs by injecting two trainable low-rank matrices into the dense layers of LLMs while keeping the original parameters frozen. However, this plug-in mechanism typically operates with a single set of parameters for all tasks, thereby restricting cross-task generalization [20, 26]. Huang et al. [20] introduced LoRAHub, a framework that composes LoRA adapters from upstream tasks to generalize to new tasks by gradient-free optimization of weight coefficients using few-shot

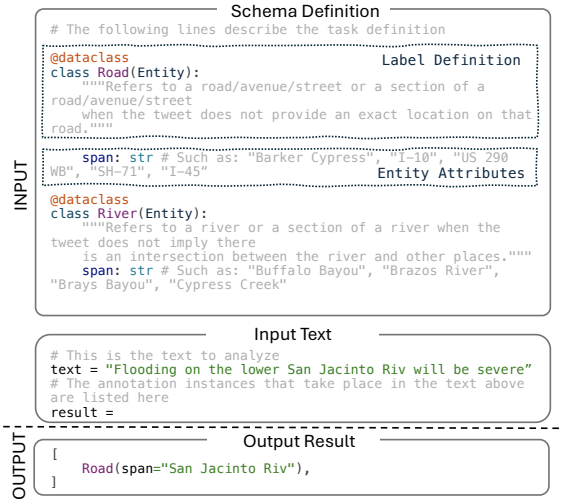


**Figure 2.** An example of NER prompt format

target examples. Similarly, Wu et al. [58] proposed to use a trainable gating mechanism to combine pretrained LoRA adapters at each layer. Furthermore, the integration of the MoE architecture [22, 47] with LoRA allows the training of a single generalizable model without additional adjustment, facilitating simultaneous multi-task learning [61]. This architecture leverages a gating mechanism to dynamically assign weights to each LoRA expert based on the input tokens, computing the output either as a weighted average of all LoRA adapters (dense gating), the top k experts with the highest weights (sparse gating) or those exceeding an activation threshold (adaptive gating) [26, 34, 51, 9, 43, 11, 39]. Recent advances also suggest that providing additional expert activation signals can further enhance multi-task gating. Dou et al. [5] and Liu et al. [33] proposed task identifiers across datasets to partition expert groups or serve as gating input. Meanwhile, Gou et al. [12] and Liu et al. [36] proposed using cluster information derived from training set as semantic task signals to activate different task expert. However, these methods often rely on predefined expert groups or cluster-derived information which operate at the group level and thus lack finer task-specific granularity. Most recently, Liao et al. [31] introduced a trainable task encoder designed to derive task embeddings from token representations and hierarchically fuse token-level and task-level information. Similarly, as such task information is typically derived from the training corpus or learned jointly with the training annotations, it may embed training-specific biases, thereby limiting adaptability and generalization to unseen or underrepresented tasks. To date, no existing research has investigated the use of multiple LoRA adapters for improving cross-task generalization in UIE tasks – something we explore in this work. Moreover, differently from the aforementioned MoE approaches, we propose leveraging training dataset-agnostic static task knowledge (encoded in the UIE instructions), combined with the token-level information, to enhance expert weighting for unseen IE tasks.

## 3 Task-Aware MoELoRA for UIE

In this section, we describe the proposed approach, starting with the applied UIE framework and followed by a detailed explanation of the MoELoRA architecture and the task-aware gating mechanism.

### 3.1 UIE Framework

We adopt the schema-based UIE framework proposed by Sainz et al. [45], and use a fine-tuned LLM to perform multiple IE tasks. To be

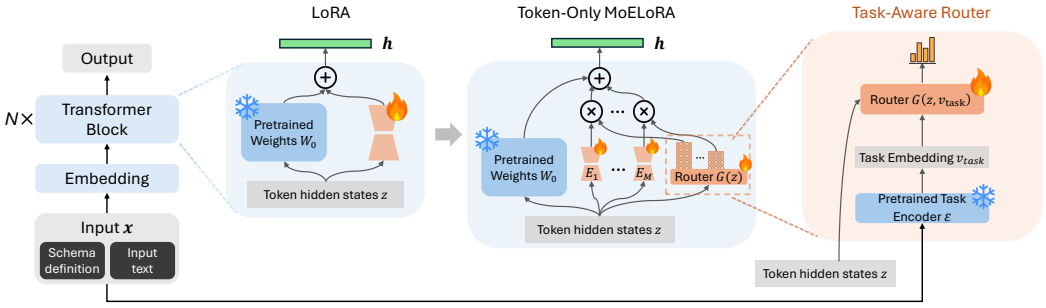


**Figure 3.** The architecture of Task-Aware MoELoRA, compared with LoRA and MoELoRA.

able to use the same framework for different IE tasks and datasets, we represent both inputs and outputs as Python classes. We show a representative example in Figure 2, illustrating the main components of the input prompt and the output for a named entity recognition (NER) task. The prompt has two components: the schema definition and the input text.

**Schema definition:** represents the IE task to be completed. Each label which can be retrieved is associated with a Python class. Class attributes specify the particular pieces of information to be extracted for each label. For instance, in the example NER task, every class has a single attribute representing the text span that corresponds to the entity. This schema definition contains comprehensive task instructions as comments, providing detailed descriptions of labels and representative examples for both labels and attributes. These instructions can be encoded to effectively capture semantic task signals. This component of the prompt allows the framework to be applied to unseen labels/datasets/tasks. In order to apply it for unseen tasks, it is enough to define a suitable schema.

Following the instructions in the schema definition, the framework analyses the text, and extracts a list of instances of the pre-defined classes, as illustrated in Figure 2.

## 3.2 Task-Aware MoELoRA Architecture

The objective of our proposed model is to dynamically adjust parameter sets for unseen tasks by leveraging similarities with previously learned tasks. Previous UIE approaches based on LLMs [45, 30] apply a single LoRA adapter to each transformer block in the LLM architecture as a fine-tuning method. However, we argue that using multiple adapters (where each adapter has its own knowledge) can provide the model with better generalization capabilities. Therefore, as shown in Figure 3, we substitute the LoRA adapter by a mixture of LoRA experts (MoELoRA).

For each transformer block, MoELoRA augments the pre-trained weights of the linear layer, which we denote as $W_0 \in \mathbb{R}^{d_{in} \times d_{out}}$. Consistent with the standard LoRA tuning approach [19], during the training phase, $W_0$ remains frozen and does not receive any gradient upgrades, while the weights in the MoELoRA component are updated. MoELoRA is divided in two sub-components: the experts, and a router.

### 3.2.1 Experts

Instead of a single adapter, each transformer block contains $M$ smaller LoRA adapters – denoted as experts. Following [19], the forward layer of each expert, $E_i$ is defined as:

$$E_i = B_i A_i z \tag{1}$$

where $A_i \in \mathbb{R}^{d_{in} \times r}$, $B_i \in \mathbb{R}^{r \times d_{out}}$ are trainable matrices, and $z \in R^{d_{in}}$ represents the token hidden state embeddings (i.e. the input vector to the transformers linear layer). Here, the rank $r$ is significantly smaller than $\min(d_{in}, d_{out})$ to substantially reduce the number of trainable parameters.

### 3.2.2 Router

To decide which experts have the best knowledge for a task, there is a router responsible for assigning appropriate weights to each expert based on the input of the transformers block. We can define the router as a function $G(x, z)$ where $x$ is the input prompt, and $z$ the token embeddings (as defined in Section 3.2.1). The router computes a weight distribution across the experts for each input token. Hence, the output vector of the transformers block, $h \in \mathbb{R}^{d_{out}}$ is computed:

$$h = W_0 z + \sum_{i=1}^{M} G(x, z)_i E_i(z) \tag{2}$$

**Token-Only router:** As a baseline router configuration for MoELoRA, we only use the token hidden states $z$ to weight the experts. This router uses a trainable linear layer $W_g \in \mathbb{R}^{d_{in} \times M}$ to compute a weight distribution across the experts for each input token, normalized using a softmax function:

$$G(x, z) = \text{Softmax}(z W_g) \tag{3}$$

**Task-Aware Router:** Using only the token embeddings to choose among experts can limit the model's ability to identify task-level characteristics – therefore hurting the generalization capabilities of MoELoRA. Ideally, for effective multi-task learning in UIE, expert allocation should consider both token-level and task-specific features. Therefore, we propose a second router configuration to avoid this problem. As depicted in Figure 2, each prompt comprises schema definitions with information extraction guidelines paired with the corresponding input text to analyse—explicitly representing the overall task objective. Based on this, we define explicit task embeddings as the encoded representations of the prompt. For computational efficiency, we use a pre-trained lightweight code encoder as our encoder, denoted as $\mathcal{E}$. The embedding $v_{task} \in \mathbb{R}^{d_{task}}$ is then obtained as:

$$v_{task}(x) = \text{avg}\left(\mathcal{E}(x_j)\right)_{j=1}^{j=|x|} \tag{4}$$

where $\mathcal{E}(x_j) \in \mathbb{R}^{d_{task}}$ represents the embedding for the $j$-th token in the task instruction $x$. This mean pooling operation provides a compact and informative representation of the task semantics. As it only depends on the prompt, the encoder (and therefore, the task embedding $v_{task}$) is shared among all the transformers blocks. To

perform task-aware routing, we concatenate the task representation $v_{task}(x)$ with the input token embeddings $z$. The router can be defined as:

$$G(x, z) = \text{Softmax}([z; v_{task}(x)]W_g) \quad (5)$$

with $W_g \in \mathbb{R}^{(d_{in}+d_{task})\times M}$. This operation allows the gating function to incorporate task-specific contextual information for dynamic expert weighting.

## 4 Experimental Setup

### 4.1 Datasets

This study focuses on evaluating the implementation of Task-Aware MoELoRA versus LoRA training, specifically in terms of generalization to unseen tasks. We also contrast the performance of UIE for seen (appear in training) and unseen (never seen before) tasks. For a fair comparison, we adopt the training corpus from Sainz et al. [45] and expand the set of unseen tasks to enable a more comprehensive analysis[1]. An overview of the datasets used in this study is provided in Figure 4[2]. Datasets marked in yellow represent the training sets, which includes 5 different tasks: Named Entity Recognition (NER), Relation Extraction (RE), Event Extraction (EE), Event Argument Extraction (EAE), and Slot Filling (SF). Datasets marked in grey indicate unseen datasets that are used exclusively for evaluation. This analysis uses only a subset of the datasets where unseen tasks exist.

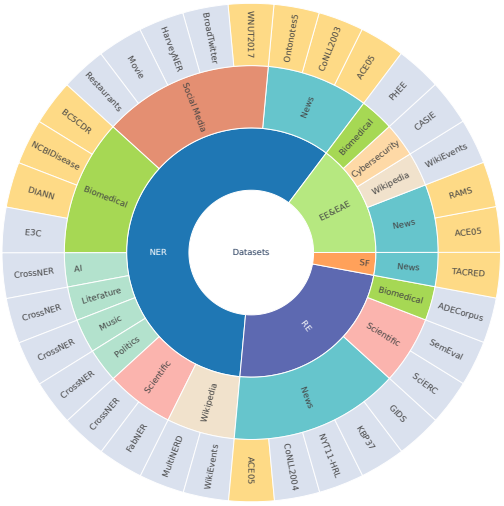


**Figure 4.** An overview of the datasets used, with the grey color representing the unseen datasets.

### 4.2 Implementation Details

**Backbone Pre-trained Models:** We adopted the 7B version Code Llama [44] as our LLM backbone, as it aligns with our objective of using code to represent both inputs and outputs. This choice also ensures a fair comparison, as the baseline framework, GoLLIE, is built on the same model [45]. For the encoder, we use the CodeT5 encoder [55], which incorporates an identifier-aware pre-training objective. This objective enhances representation learning by leveraging meaningful semantic identifiers in the code, where, in our prompt, the identifier specifically refers to the assigned Label class.

[1] Test splits are adopted from IEPILE[13]

[2] Detailed dataset statistics are provided in the appendix. The appendix and code are available at https://github.com/lubingzhiguo/TA-MoELoRA

**Baselines:** LoRA serves as our primary benchmark, as it allows us to evaluate the direct impact of introducing a router while maintaining consistent training settings. To ensure uniformity across all PEFT methods, we configured the global rank and alpha to 32, with LoRA at rank 32 serving as the baseline. Additionally, to mitigate the parameter-level bias resulting from the router's additional parameters, we include a LoRA variant with a slightly larger rank of 36 to ensure a fair comparison. Since our method is designed for zero-shot scenarios, we also compare its performance with several relevant baselines: our base framework [45], a similar code-style UIE model [30], and an LLM-based Universal NER model [64].

**Model Configurations:** In the MoELoRA method, ranks are evenly distributed across all experts, with each expert assigned a rank of $32/M$. In this study, we use $M = 8$ experts, each assigned a rank $r = 4$, and compare three versions of the MoELoRA method. The classic variant, *Token-Only*, uses the token as the gating input. Two task-aware variants are also evaluated: *Schema-Aware*, where only the schema definition is provided to the task encoder, and *Task-Aware*, which uses the full inference prompt, including schema and input text as task definition.

**Training Setup:** All adapters are integrated into the linear layers, including the $q, k, o, v$ projections in attention layers and $up, down, gate$ layers in feed-forward network. All models were trained for 3 epochs with an effective batch size of 32 and a learning rate of 3e-4 with a cosine scheduler. Training and inference were conducted on two NVIDIA RTX 4090 GPUs, using a fixed global seed of 42.

## 5 Results

We evaluate the impact of the proposed task-aware method on the performance of UIE tasks. In particular, we investigate the following research questions:

**RQ1:** How does the use of MoELoRA affect the effectiveness of UIE models?

**RQ2:** How does performance of task-aware router compare to token-level gating on unseen datasets?

### 5.1 MoELoRA Performance

We begin by examining the first component of this work: how does the use of multiple LoRA experts with a router compare to classic LoRA in terms of its impact on the effectiveness of the UIE model? In particular, we are interested in assessing whether MoELoRA offers more benefits for unseen tasks.

We compare LoRA to MoELoRA in two settings, using the F1 metric: first, a supervised setting, where the performance is tested on the evaluation sets of the datasets used for training the models; then, a zero-shot setting, where we test our method on the test set of unseen datasets. Results are shown in Table 1. As we want to measure the effectiveness of the mixture of experts, we first compare the performance of LoRA (first two columns) with Token-Only MoELoRA (third column).

**Overall performance:** As shown in Table 1, Token-Only MoELoRA performs slightly worse than LoRA in supervised settings, showing a decrease in overall performance of approximately 1.5%. However, MoELoRA often significantly outperforms or matches the performance of LoRA across various datasets in zero-shot settings. This improvement is consistent across all three major information extraction tasks: NER, RE, EE&EAE with an overall increase of 4.5% in F1 scores in zero-shot scenarios. This differential performance indicates

**Table 1.** Results overview. Best F1 value for each dataset is highlighted in bold. † denotes significant improvements (t-test $p < 0.05$) with respect to both LoRA variants and $*$ with respect to Token-Only MoELoRA. Arrows (↑ and ↓) indicate comparisons with the best-performing LoRA variant.

| PEFT Method | LoRA | | MoELoRA | | |
|---|---|---|---|---|---|
| Variant | rank 32 | rank 36 | Token-Only | Schema-Aware | Task-Aware |
| Params | ~80M | ~90M | ~89M | ~90M | ~90M |
| **Supervised Setting** | | | | | |
| $ACE05_{NER}$ [53] | **87.7** | 87.5 | 81.9 | 77.6 | 83.1 |
| $ACE05_{RE}$ [53] | **58.8** | 57.0 | 56.4 | 48.2 | 54.7 |
| $ACE05_{EE}$ [53] | 69.2 | 69.0 | 65.7 | 62.8 | **69.3** |
| $ACE05_{EAE}$ [53] | **65.0** | 64.6 | 57.5 | 51.9 | 55.0 |
| CoNLL 2003 [52] | **92.6** | **92.6** | 92.4 | 91.9 | 91.7 |
| Ontonotes 5 [42] | 70.7 | 71.8 | **80.5** | 80.0 | 79.7 |
| WNUT 2017 [4] | 51.6 | 49.3 | 51.7 | 50.0 | **53.4** |
| BC5CDR [57] | **87.3** | 87.3 | 86.0 | 83.2 | 85.8 |
| NCBIDisease [21] | **86.6** | 86.2 | 84.9 | 83.9 | 84.3 |
| DIANN [7] | 80.5 | **80.9** | 78.1 | 77.2 | 77.8 |
| RAMS [6] | 47.6 | **48.4** | 45.3 | 44.2 | 43.9 |
| TACRED [63] | **57.2** | 56.7 | 56.2 | 54.7 | 56.8 |
| Average | **71.2** | 70.9 | $69.7^{\downarrow 1.5\%}$ | $67.1^{\downarrow 4.1\%}$ | $69.6^{\downarrow 1.6\%}$ |
| **Zero-shot Setting** | | | | | |
| **NER** | | | | | |
| BroadTwitter [3] | 47.2 | 46.7 | $48.7^{\dagger}$ | $49.2^{\dagger *}$ | $\mathbf{50.2}^{\dagger *}$ |
| HarveyNER [2] | 38.5 | 35.7 | 36.8 | $\mathbf{41.3}^{\dagger *}$ | 36.7 |
| AI [35] | 54.7 | 50.3 | $57.5^{\dagger}$ | $\mathbf{58.8}^{\dagger}$ | $57^{\dagger}$ |
| Literature [35] | 61.6 | 62.1 | 64.0 | $67.1^{\dagger *}$ | $\mathbf{67.5}^{\dagger *}$ |
| Music [35] | 64.1 | 62.6 | $72.4^{\dagger}$ | $\mathbf{74.2}^{\dagger}$ | $72.7^{\dagger}$ |
| Politics [35] | 61.7 | 55.5 | 62.2 | $\mathbf{66.0}^{\dagger *}$ | 58.6 |
| Science [35] | 53.0 | 49.1 | $55.3^{\dagger}$ | $\mathbf{55.5}^{\dagger}$ | 52.0 |
| FabNER [25] | **25.4** | 22.7 | 24.7 | 24.5 | $25.3^{*}$ |
| MIT Movie [32] | 60.3 | 61.4 | $63.9^{\dagger}$ | $62.7^{\dagger *}$ | $\mathbf{64.7}^{\dagger *}$ |
| MIT Restaurants [32] | 41.7 | 41.6 | $47.5^{\dagger}$ | $48.4^{\dagger}$ | $\mathbf{52.6}^{\dagger *}$ |
| E3C [40] | 57.8 | 56.4 | $61.6^{\dagger}$ | $60.6^{\dagger}$ | $\mathbf{62.7}^{\dagger *}$ |
| MultiNERD [50] | 74.5 | 75.8 | 75.9 | 75.5 | $\mathbf{77.2}^{\dagger *}$ |
| WikiEvents [28] | **81.2** | 80.6 | 77.5 | 74.2 | 75.3 |
| Average | 55.5 | 53.9 | $57.5^{\uparrow 2\%}$ | $\mathbf{58.3}^{\uparrow 2.8\%}$ | $57.9^{\uparrow 2.4\%}$ |
| **RE** | | | | | |
| SciERC [38] | 0.2 | 0.8 | $5.4^{\dagger}$ | $2.4^{\dagger}$ | $\mathbf{13.7}^{\dagger *}$ |
| SemEval [18] | 24.0 | 24.6 | $38.3^{\dagger}$ | $37.0^{\dagger}$ | $\mathbf{53.4}^{\dagger *}$ |
| CoNLL 2004 [1] | 20.2 | 21.2 | $38.7^{\dagger}$ | $53.3^{\dagger *}$ | $\mathbf{59.0}^{\dagger *}$ |
| NYT11-HRL [49] | 15.1 | 17.2 | 18.9 | $18.2^{\dagger}$ | $\mathbf{21.6}^{\dagger *}$ |
| KBP37 [62] | 9.5 | 16.1 | $22.6^{\dagger}$ | $22.8^{\dagger}$ | $\mathbf{25.4}^{\dagger *}$ |
| GIDS [23] | 54.4 | 62.7 | $68.5^{\dagger}$ | $77^{\dagger *}$ | $\mathbf{79.7}^{\dagger *}$ |
| ADECorpus [15] | 24 | 24.6 | $38.3^{\dagger}$ | $37^{\dagger}$ | $\mathbf{53.4}^{\dagger *}$ |
| Average | 18.6 | 22.3 | $29.6^{\uparrow 7.3\%}$ | $32.7^{\uparrow 10.4\%}$ | $\mathbf{40.1}^{\uparrow 17.8\%}$ |
| **EE & EAE** | | | | | |
| $WikiEvents_{EE}$ [28] | 46.0 | **46.4** | 44.3 | 46.1 | 41.9 |
| $CASIE_{EE}$ [46] | 48.8 | 50.7 | $56.5^{\dagger}$ | $61.6^{\dagger *}$ | $\mathbf{68}^{\dagger *}$ |
| $PHEE_{EE}$ [48] | 55.7 | 57.2 | $63.3^{\dagger}$ | $67.1^{\dagger *}$ | $\mathbf{70.3}^{\dagger *}$ |
| $WikiEvents_{EAE}$ [28] | 50.1 | **50.7** | 49.4 | 48.7 | 46.7 |
| $CASIE_{EAE}$ [46] | 48.5 | 47.3 | 49.2 | $\mathbf{51.3}^{\dagger *}$ | 47.7 |
| $PHEE_{EAE}$ [48] | 56.2 | 54.3 | $62.1^{\dagger}$ | $65.5^{\dagger *}$ | $\mathbf{67.1}^{\dagger *}$ |
| Average | 50.9 | 51.1 | $54.1^{\uparrow 3\%}$ | $56.7^{\uparrow 5.6\%}$ | $\mathbf{57.0}^{\uparrow 5.9\%}$ |
| Average All | 44.5 | 44.7 | $49.2^{\uparrow 4.5\%}$ | $51.0^{\uparrow 6.3\%}$ | $\mathbf{52.9}^{\uparrow 8.2\%}$ |

that, while MoELoRA sacrifices some accuracy in supervised scenarios, it is particularly effective in adapting to new and unseen data. Moreover, we see that this is not an artifact of the router adding more parameters, as the equivalently-sized rank-36 LoRA variant still underperforms MoELoRA.

**Performance on unseen labels:** To further evaluate the generalization capabilities of MoELoRA in handling unseen data, we partition the labels in the zero-shot datasets based on whether a particular label was present or not in the training datasets. This is motivated by the observation [45] that some labels in the zero-shot datasets may conceptually overlap with those in the training corpus, despite differences in annotation guidelines and naming conventions. We report the averaged F1 score for all tasks on both seen and unseen labels (unexposed during training) in Table 2. Overall, the Token-Only MoELoRA model demonstrates performance improvements across both label sets. Importantly, the gains are more considerable for unseen labels, with a 5.1% increase in F1 score, where there is a 2.1% higher than the 3% gain observed for seen labels. This demonstrates

**Table 2.** Averaged F1 for seen and unseen labels in the zero-shot setting across tasks. Highest values are highlighted in bold. Arrows (↑ and ↓) indicate comparisons with the best-performing LoRA variant.

| PEFT Method | LoRA | | MoELoRA | | |
|---|---|---|---|---|---|
| Variant | rank 32 | rank 36 | Token-Only | Schema-Aware | Task-Aware |
| **Seen Labels** | | | | | |
| NER | 57.6 | 54.9 | 59.8 | 60.1 | **61.3** |
| RE | 24.9 | 32.3 | 33.3 | 38.6 | **44.7** |
| EE&EAE | 37.7 | 38.1 | **38.3** | 37.5 | 37.2 |
| Average All | 45.0 | 45.7 | $48.7^{\uparrow 3\%}$ | $50.1^{\uparrow 4.4\%}$ | $\mathbf{52.4}^{\uparrow 6.7\%}$ |
| **Unseen Labels** | | | | | |
| NER | 46.0 | 44.0 | 49.3 | **51.5** | 48.7 |
| RE | 14.2 | 15.6 | 25.5 | 27.2 | **35.8** |
| EE&EAE | 40.7 | 40.4 | 43.9 | **48.4** | 48.0 |
| Average All | 37.3 | 36.5 | $42.4^{\uparrow 5.1\%}$ | $45.1^{\uparrow 7.8\%}$ | $\mathbf{45.6}^{\uparrow 8.3\%}$ |

its potential to enhance the robustness of UIE models when applied to diverse datasets, particularly in low-resource or zero-shot scenarios where adaptability is crucial.

**To answer RQ1:** Replacing LoRA with MoELoRA structure can robustly improve the performance on unseen datasets, especially for unseen labels. Nonetheless, this approach results in a marginal performance reduction in supervised settings. In low-resource scenarios, where annotations are unavailable, MoELoRA presents an effective strategy for enhancing out-of-domain performance by leveraging knowledge from diverse datasets.

## *5.2 Task-Aware MoELoRA Performance*

To investigate the generalization ability of task-aware routing, we compare performance of task-aware routers to token gating, with a special focus on unseen datasets.

**Token-only vs. Schema-aware routing.** We show in Tables 1 and 2 the comparison between the Token-Only MoELoRA (column 3) and a model with a task-aware router that considers the schema definition (Schema-Aware, column 4). As illustrated in Table 1, although the schema-aware variant reduces supervised performance by 2.6%, it surpasses Token-Only MoELoRA on unseen datasets, achieving an overall improvement of 1.8%. Table 2 further shows the effectiveness of Schema-Aware MoELoRA at extracting information from both seen and unseen labels (achieving, respectively, 1.4% and 2.7% improvements over the Token-Only variant), showing the importance of combining task instructions with token information to weight experts.

**Impact of task definition:** We further investigate how the task representation impacts generalization performance, by comparing the Schema-Aware MoELoRA with a variant which uses the same architecture, but also provides the input text to the task encoder as additional information. We denote this variant as Task-Aware MoELoRA. As seen in Table 1, in supervised settings, the Task-Aware Variant outperforms the Schema-Aware variant by 2.5% and demonstrates performance comparable to LoRA variants. Moreover, it surpasses both the Schema-Aware and Token-Only variants in zero-shot settings and consistently excels at relationship extraction. Notably, it significantly outperforms the token-only router model in 17 out of 26 zero-shot datasets (65.4%), 5 more than the Schema-Aware variant. For the RE task, we observe the largest improvements, with the Task-Aware MoELoRA outperforming Schema-Aware variants by 7.4% and LoRA by 17.8%. This is particularly noteworthy given that RE tasks constitute the smallest portion of the training datasets, where NER dominates (refer to Figure 4). Similarly, Task-Aware variant achieves the highest averaged performance across both seen and unseen label sets, outperforming Schema-Only MoELoRA by 2.3% for

**Table 3.** Comparison of UIE models under zero-shot setting. Best F1 values are highlighted in bold.

| | UniNER | KnowCoder | GoLLIE | Ours |
|---|---|---|---|---|
| AI | **62.3** | 60.3 | 59.1 | 57 |
| Literature | 67.4 | 61.1 | 62.7 | **67.5** |
| Music | 69 | 70 | 67.8 | **72.7** |
| Politics | 64.5 | **72.2** | 57.2 | 58.6 |
| Science | **66.9** | 59.1 | 55.5 | 52 |
| Movie | 54.2 | 50 | 63 | **64.7** |
| Restaurants | 16 | 48.2 | 43.4 | **52.6** |
| GIDS | - | 25.5 | 64.3 | **79.7** |
| $\text{CASIE}_{\text{EE}}$ | - | 58.2 | 55.1 | **68** |
| Average | - | 56.1 | 58.7 | **63.6**$^{\uparrow 4.9\%}$ |

seen labels and 0.5% for unseen labels (and by 6.1% for seen labels, and 8.6% for seen labels for the RE task). These observations suggest that incorporating the input text along with the schema to the gating mechanism further improves the generalization of our model to unseen tasks (particularly, for the RE task), while maintaining performance on seen datasets.

**Analysis of failure cases**: While the above analysis focuses on the average performance across broad tasks, we also explore dataset-specific behavior under in zero-shot settings. Interestingly, although MoELoRA variants generally outperform or match LoRA variants on most datasets, they consistently underperform on one particular dataset: WikiEvents. This discrepancy can be attributed to the nature of the dataset, which is constructed from Wiki-based sources [28] that overlap with many training datasets. As a result, most labels in this dataset were already seen during training, with only approximately 5% of annotated labels in WikiEvents (174 out of 3,374, each instance may have multiple annotations) remaining unseen. This overlap shifts the evaluation from a zero-shot to a more supervised-like setting, thereby aligning with the earlier observation that MoELoRA variants tend to exhibit reduced performance in such conditions. Moreover, the observed performance gap can be explained by the architectural design of MoELoRA, which distributes its parameter space across multiple experts. This modular structure allows the model to encode learned evidence more evenly across various tasks/datasets, introducing an implicit regularization effect that helps prevent overfitting and thus further supports generalization. However, this benefit can become a limitation in fully supervised settings, where abundant training data from the target tasks is available. In such cases, emphasizing information from the target tasks during training becomes more important—an objective better fulfilled by a unified parameter-set, as implemented in basic LoRA.

**Comparison with other UIE methods:** We further compare the Task-Aware variant with other LLM-based UIE models, including the 7B versions of UniNER [64], KnowCoder [30], and GoLLIE [45]. This analysis aims to evaluate the effectiveness of the proposed method against existing approaches in the field. For this experiment we report the same zero-shot datasets as KnowCoder [30], to provide a fair comparison. Unlike the previous experiments this data follows a different task distribution, i.e. it is largely composed of NER tasks (7/9). However, certain biases remain, as the models (excluding GoLLIE) are trained on different datasets, which is not accounted for in the evaluation setup of the proposed method. The comparison results in zero-shot settings are presented in Table 3. It can be seen that our model surpasses the performance on most datasets, achieving a success rate of 57.1% (4/7) compared to UniNER, 66.7% (6/9) compared to KnowCoder, and 77.8% (7/9) compared to our base framework, GoLLIE, with an average improvement of 4.9%. Notably, this is achieved despite evaluation configurations favoring NER tasks and training from a slightly weaker starting point, as evidenced by the performance of reproduced LoRA compared to the reported results.

**To answer RQ2:** Routers that consider both task-level and token-level information can significantly improve the token routing performance on unseen datasets, especially for tasks with limited exposure during training. Task-Aware MoELoRA, which considers both schema and contextual text as task representations, outperforms Token-Only MoELoRA in zero-shot settings with only approximately 1M additional parameters while maintaining comparable performance in supervised settings.

## *5.3 Ablation Study*

**Effect of Expert Count:** To validate the impact of the number of experts on effectiveness in zero-shot datasets, we compare the results obtained by varying the number of experts (2, 4, and 16) while maintaining a global rank of 32. Figure 5 illustrates varying trends in the performance metrics across the three tasks. In particular, RE tasks show steady improvements as the number of experts increases, with peak performance observed at 8 experts. Therefore, considering both zero-shot performance and training time efficiency, 8 experts is the optimal choice for a balanced performance across tasks.

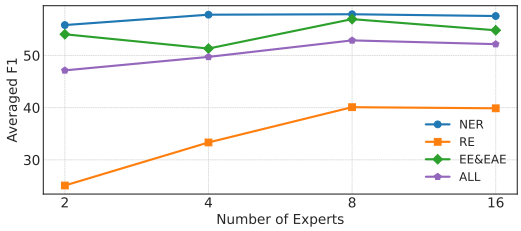


**Figure 5.** Zero-shot performance by number of experts

**Effects of Router Design:** In this work, we focus on a dense design to leverage all experts. We further investigate another common router design, namely the sparse gate [47], which selects the top-rated experts for computation. This mechanism improves efficiency by activating only the most suitable experts, thereby reducing computational costs. For instance, the sparse gate for the Token-Only variant can be formulated as follows:

$$G(x, z) = \text{Softmax}\left(\text{Top-K}(zW_g, k)\right) \tag{6}$$

where Top-K() keeps only the top $k$ largest values. We report the results of the Top-2 gating versus dense gating comparison in Table 4. It is evident that MoELoRA with a Top-2 router consistently underperforms compared to the dense router on UIE tasks, with a notable decline in performance, especially in the Task-Aware setup. This suggests that the dense router allows the model to better leverage shared knowledge in UIE tasks by using all experts.

**Table 4.** Performance of different router designs.

| | Router | Supervised Avg. F1 | Zero-shot Avg. F1 |
|---|---|---|---|
| Token-Only | Dense | **69.7** | 49.2 |
| | Top-2 | 50.0 | 46.8 |
| Task-Aware | Dense | 69.6 | **52.9** |
| | Top-2 | 43.5 | 35.2 |

**Effects of Task Encoder:** Another important factor influencing the proposed method is the task encoder which generates contextual instruction embeddings. In this study, we primarily evaluate the performance of CodeT5 [55] as our task encoder. To further investigate the influence of different encoder models, we conduct an ablation study comparing two categories:

**Table 5.** Performance of Task-Aware MoELoRA with different task encoders. Bold numbers indicate F1 improvements over codet5.

| Encoder | Model | Supervised | Zero-shot Avg. F1 | |
|---|---|---|---|---|
| | | Avg. F1 | Seen | Unseen |
| Code | codet5 | 69.6 | 52.4 | 45.6 |
| | codebert | 56.2 | 49.4 | 45.1 |
| | graphcodebert | 68.4 | 52.0 | **47.6** |
| | codet5p-110m | **70.2** | 50.6 | 44.7 |
| | jina-embeddings | 68.7 | 51.7 | **49.0** |
| Text | bge | 69.5 | 48.5 | 42.9 |
| | gte | 57.8 | 46.6 | 43.5 |

- Code Encoder: CodeT5 [55], CodeBERT [10], GraphCodeBERT [14], CodeT5+ [56], Jina Embeddings [16].
- Text Encoder: BGE [59] and GTE [29].

This choice is motivated by the fact that task instructions are formatted in code-style while also containing comprehensive textual guidelines. Our analysis aims to provide a deeper understanding of task encoder selection. For generating task embeddings, we apply mean pooling, except in BERT-based models, where we use the embeddings of the [CLS] token. Table 5 presents a performance comparison of various code and text embedding models. The results demonstrate that code-based encoders generally outperform text encoders in zero-shot settings. Models like Jina Embeddings and GraphCodeBERT produce better results on unseen label sets, achieving F1 scores of 49.0 and 47.6, respectively. However, this enhanced zero-shot capability comes with a trade-off, often leading to reduced performance on seen labels and supervised tasks. This analysis indicates that task embeddings produced by the encoder are essential for generalization, with "higher-quality" representations potentially enhancing adaptability to unseen tasks. Furthermore, using CodeT5 as the task encoder yields robust outcomes, maintaining a balance between supervised specialization and zero-shot generalization.

### *5.4 Case Study*

**Analysis of Expert Weight Distribution:** To better understand the behavior across tasks, we further examine the routing weights associated with each task. In this context, each task corresponds to a distinct label within a dataset, aligned with the prompt-level task signal employed in this study. For better interpretability, we collected the expert weight distribution from NER task instances that are assigned a single label, ensuring clearer separation in the task space. The expert weights from both the Self-Attention and Feed-Forward Network (FFN) modules were concatenated across all layers for each instance. Figure 6 shows a t-SNE projection of learned expert weight representations corresponding to different labels. Each node is annotated with its label name, and nodes of the same color belong to the same dataset. As shown, while labels from the same dataset tend to exhibit similar weighting patterns, labels with the same meaning (e.g., Award, Event) or those with semantically similar meanings across datasets (e.g., Researcher and Scientist, Clinical Entity and Disease) also tend to cluster closely in the embedding space. Therefore, incorporating the prompt-based task embeddings as an additional signal facilitates finer-grained, label-level routing behavior.

## 6 Discussions

The effectiveness of our approach is influenced by three factors: (a) the design of task instructions, (b) the design of task encoders and (c) the backbone large language model.

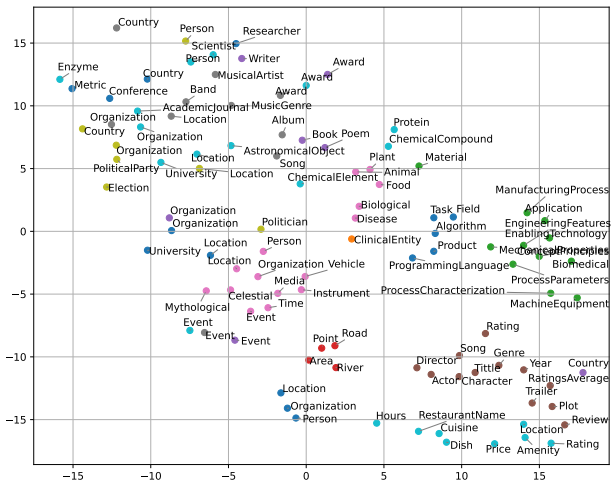


**Figure 6.** t-SNE visualization of expert weight distribution across labels

**Task Instructions**: Regarding the task instructions, we have restricted our experiments to the schema definitions provided by Sainz et al. [45] if available, or schema definitions in the same format for the newly added zero-shot datasets. While different expressions might have an effect on the performance of different UIE models, we limited our research to only these task instructions to ensure a fair comparison with the GoLLIE baseline. Future work might explore alternative schema content and formatting.

**Task Encoders**: With respect to the task encoders, we have tested seven pre-trained embedding models as input to our task-aware router, reported earlier in Table 5. The task embedding space used can have a large impact on generalization capability (as we might expect), but we have only performed a shallow comparison of popular models at this point. We believe that their may be significantly better embedding spaces than what was tested here that can better capture the commonalities between tasks, and hence lead to better expert weighting.

**Backbone Model**: We limited our experiments to a single LLM backbone: Code Llama 7B. This model was selected to ensure a fair comparison of our Task-Aware MoELoRA fine-tuning approach with GoLLIE [45]. We did not experiment with more models on cost grounds, given the large number of datasets and settings tested. If we used a larger model, we expect that overall performance will increase [45], but the pattern's observed will remain the same. However, it may be worth investigating how models with additional reasoning training/distillation perform for UIE tasks.

## 7 Conclusions

In this paper, we have introduced a new parameter efficient fine-tuning framework for UIE, Task-Aware MoELoRA, which combines multiple LoRA adapters with a gating mechanism incorporating explicit task information to decide the importance of each expert. Through empirical evaluation, we have shown that our proposed model outperforms the use of a single LoRA adapter and prior code-schema-based UIE models on unseen tasks while performing comparably on previously seen scenarios.

Our results highlight that using multiple adapters to leverage knowledge from publicly available datasets can lead to improved generalization of information extraction models in low-resource scenarios. Especially when contextual task information is used to estimate the importance of each of these adapters.

## References


[1] X. Carreras and L. Màrquez. Introduction to the CoNLL-2004 Shared Task: Semantic Role Labeling. In *Proc. of CoNLL at HLT-NAACL*, 2004.

[2] P. Chen, H. Xu, et al. Crossroads, Buildings and Neighborhoods: A Dataset for Fine-grained Location Recognition. In *Proc. of NAACL*, 2022.

[3] L. Derczynski, K. Bontcheva, et al. Broad Twitter Corpus: A Diverse Named Entity Recognition Resource. In *Proc. of COLING*, 2016.

[4] L. Derczynski, E. Nichols, et al. Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition. In *Proc. of WNUT*, 2017.

[5] S. Dou, E. Zhou, et al. LoRAMoE: Alleviating World Knowledge Forgetting in Large Language Models via MoE-Style Plugin. In *Proc. of ACL*, 2024.

[6] S. Ebner, P. Xia, et al. Multi-Sentence Argument Linking. In *Proc. of ACL*, 2020.

[7] H. Fabregat, J. Martínez-Romo, et al. Overview of the DIANN Task: Disability Annotation Task. In *Proc. of IberEval co-located with SEPLN*, 2018.

[8] W. Fedus, B. Zoph, et al. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 2022.

[9] W. Feng, C. Hao, et al. Mixture-of-LoRAs: An Efficient Multitask Tuning Method for Large Language Models. In *Proc. of (LREC-COLING)*, Torino, Italia, 2024.

[10] Z. Feng, D. Guo, et al. CodeBERT: A Pre-Trained Model for Programming and Natural Languages. In *Findings of EMNLP*, 2020.

[11] C. Gao, K. Chen, et al. MoLA: MoE LoRA with layer-wise expert allocation. In *Findings of NAACL*, 2025.

[12] Y. Gou, Z. Liu, et al. Mixture of Cluster-conditional LoRA Experts for Vision-language Instruction Tuning, 2024. arXiv preprint arXiv:2312.12379.

[13] H. Gui, L. Yuan, et al. IEPile: Unearthing Large Scale Schema-Conditioned Information Extraction Corpus. In *Proc. of ACL*, 2024.

[14] D. Guo, S. Ren, et al. GraphCodeBERT: Pre-training Code Representations with Data Flow. In *Proc. of ICLR*, 2021.

[15] H. Gurulingappa, A. M. Rajput, et al. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *Journal of Biomedical Informatics*.

[16] M. Günther, J. Ong, et al. Jina Embeddings 2: 8192-Token General-Purpose Text Embeddings for Long Documents, 2024. arXiv preprint arXiv:2310.19923.

[17] H. Hamad, A. K. Thakur, et al. FIRE: A Dataset for Financial Relation Extraction. In *Findings of NAACL*, 2024.

[18] I. Hendrickx, S. N. Kim, et al. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals. In *Proc. of SemEval*, 2010.

[19] E. Hu, Y. Shen, et al. LoRA: Low-Rank Adaptation of Large Language Models. In *Proc. of ICLR*, 2022.

[20] C. Huang, Q. Liu, et al. LoraHub: Efficient Cross-Task Generalization via Dynamic LoRA Composition. In *Proc. of COLM*, 2024.

[21] R. Islamaj Doğan et al. An improved corpus of disease mentions in PubMed citations. In *Proc. of BioNLP*, 2012.

[22] R. A. Jacobs, M. I. Jordan, et al. Adaptive Mixtures of Local Experts. *Neural Computation*, 1991.

[23] S. Jat, S. Khandelwal, et al. Improving distantly supervised relation extraction using word and entity based attention. In *Proc. of AKBC co-located with NeurIPS*, 2017.

[24] V. Kocaman and D. Talby. Accurate Clinical and Biomedical Named Entity Recognition at Scale. *Software Impacts*, 2022.

[25] A. Kumar and B. Starly. "FabNER": information extraction from manufacturing process science domain literature using named entity recognition. *Journal of Intelligent Manufacturing*, 2022.

[26] D. Li, Y. Ma, et al. MixLoRA: Enhancing Large Language Models Fine-Tuning with LoRA-based Mixture of Experts, 2024. arXiv preprint arXiv:2404.15159.

[27] P. Li, T. Sun, et al. CodeIE: Large Code Generation Models are Better Few-Shot Information Extractors. In *Proc. of ACL*, 2023.

[28] S. Li, H. Ji, et al. Document-Level Event Argument Extraction by Conditional Generation. In *Proc. of NAACL*, 2021.

[29] Z. Li, X. Zhang, et al. Towards General Text Embeddings with Multi-stage Contrastive Learning, 2023. arXiv preprint arXiv:2308.03281.

[30] Z. Li, Y. Zeng, et al. KnowCoder: Coding Structured Knowledge into LLMs for Universal Information Extraction. In *Proc. of ACL*, 2024.

[31] M. Liao, W. Chen, et al. HMoRA: Making LLMs more effective with hierarchical mixture of loRA experts. In *Proc. of ICLR*, 2025.

[32] J. Liu, P. Pasupat, et al. Asgard: A portable architecture for multilingual dialogue systems. In *Proc. of ICASSP*, 2013.

[33] Q. Liu, X. Wu, et al. When MOE Meets LLMs: Parameter Efficient Fine-tuning for Multi-task Medical Applications. In *Proc. of SIGIR*, 2024.

[34] Z. Liu and J. Luo. AdaMoLE: Fine-Tuning Large Language Models with Adaptive Mixture of Low-Rank Adaptation Experts. In *Proc. of COLM*, 2024.

[35] Z. Liu, Y. Xu, et al. Crossner: Evaluating cross-domain named entity recognition. In *Proc. of AAAI*, 2021.

[36] Z. Liu, K. Chen, et al. Task-customized Masked Autoencoder via Mixture of Cluster-conditional Experts. In *Proc. of ICLR*, 2023.

[37] Y. Lu, Q. Liu, D. Dai, et al. Unified Structure Generation for Universal Information Extraction. In *Proc. of CoNLL*, 2022.

[38] Y. Luan, L. He, et al. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proc. of EMNLP*, 2018.

[39] T. Luo, J. Lei, et al. Moelora: Contrastive learning guided mixture of experts on parameter-efficient fine-tuning for large language models. 2024. arXiv preprint arXiv:2402.12851.

[40] B. Magnini, B. Altuna, et al. The e3c project: European clinical case corpus. In *Proc. of SEPLN*, 2021.

[41] S. Mangrulkar, S. Gugger, et al. PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods. https://github.com/huggingface/peft, 2022.

[42] S. Pradhan, A. Moschitti, et al. Towards Robust Linguistic Analysis using OntoNotes. In *Proc. of CoNLL*, 2013.

[43] P. Qing, C. Gao, et al. AlphaLoRA: Assigning LoRA Experts Based on Layer Training Quality. In *Proc. of EMNLP*, 2024.

[44] B. Rozière, J. Gehring, et al. Code Llama: Open Foundation Models for Code, 2024. arXiv preprint arXiv:2308.12950.

[45] O. Sainz, I. García-Ferrero, et al. GoLLIE: Annotation Guidelines improve Zero-Shot Information-Extraction. In *Proc. of ICLR*, 2024.

[46] T. Satyapanich, F. Ferraro, et al. CASIE: Extracting Cybersecurity Event Information from Text. In *Proc. of AAAI*, 2020.

[47] N. Shazeer, A. Mirhoseini, et al. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *Proc. of ICLR*, 2017.

[48] Z. Sun, J. Li, et al. PHEE: A Dataset for Pharmacovigilance Event Extraction from Text. In *Proc. of EMNLP*, 2022.

[49] R. Takanobu, T. Zhang, et al. A Hierarchical Framework for Relation Extraction with Reinforcement Learning. In *Proc. of AAAI*, 2019.

[50] S. Tedeschi and R. Navigli. MultiNERD: A Multilingual, Multi-Genre and Fine-Grained Dataset for Named Entity Recognition (and Disambiguation). In *Findings of NAACL*, 2022.

[51] C. Tian, Z. Shi, et al. HydraLoRA: An Asymmetric LoRA Architecture for Efficient Fine-Tuning. In *Proc. of NeurIPS*, 2024.

[52] E. F. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proc. of CoNLL at HLT-NAACL*, 2003.

[53] C. Walker, S. Strassel, et al. ACE 2005 Multilingual Training Corpus, 2006. LDC Catalog No. LDC2006T06.

[54] X. Wang, W. Zhou, et al. InstructUIE: Multi-task Instruction Tuning for Unified Information Extraction, 2023. arXiv preprint arXiv:2304.08085.

[55] Y. Wang, W. Wang, et al. CodeT5: Identifier-aware Unified Pre-trained Encoder-Decoder Models for Code Understanding and Generation. In *Proc. of EMNLP*, 2021.

[56] Y. Wang, H. Le, et al. CodeT5+: Open code large language models for code understanding and generation. In *Proc of EMNLP*, 2023.

[57] C.-H. Wei, Y. Peng, et al. Assessing the state of the art in biomedical relation extraction: overview of the BioCreative V chemical-disease relation (CDR) task. *Database*, 2016.

[58] X. Wu, S. Huang, et al. Mixture of LoRA Experts. In *Proc. of ICLR*, 2024.

[59] S. Xiao, Z. Liu, et al. C-Pack: Packed Resources For General Chinese Embeddings. In *Proc. of SIGIR*, 2024.

[60] D. Ye, Y. Lin, et al. Packed Levitated Marker for Entity and Relation Extraction. In *Proc. of ACL*, 2022.

[61] T. Zadouri, A. Üstün, et al. Pushing Mixture of Experts to the Limit: Extremely Parameter Efficient MoE for Instruction Tuning. In *Proc. of ICLR*, 2024.

[62] D. Zhang and D. Wang. Relation Classification via Recurrent Neural Network, 2015. arXiv preprint arXiv:1508.01006.

[63] Y. Zhang, V. Zhong, et al. Position-aware Attention and Supervised Data Improve Slot Filling. In *Proc. of EMNLP*, 2017.

[64] W. Zhou, S. Zhang, et al. UniversalNER: Targeted Distillation from Large Language Models for Open Named Entity Recognition. In *Proc. of ICLR*, 2024.