Semi-Supervised Relation Extraction Informed by Area Under the Margin Ranking and Large Language Models

Nikita Gautam
Department of Computer Science
Kansas State University
Manhattan, Kansas, USA
ngautam@ksu.edu

Bipin Paudel
Department of Computer Science
Kansas State University
Manhattan, Kansas, USA
bipinp@ksu.edu

Doina Caragea
Department of Computer Science
Kansas State University
Manhattan, Kansas, USA
dcaragea@ksu.edu

Cornelia Caragea
Department of Computer Science
University of Illinois Chicago
Chicago, IL, USA
cornelia@uic.edu

***Abstract*—Relation extraction is an important task for understanding relationships between entities, building knowledge graphs, and facilitating knowledge discovery. Pre-trained models can be fine-tuned for relation extraction if a substantial amount of labeled data is available. However, acquiring extensive labeled data is generally challenging. Semi-supervised techniques for low-resource relation extraction, such as self-training, offer a promising solution by leveraging both limited labeled data and vast unlabeled data to mitigate this challenge. Traditional self-training methods use a teacher-student framework, where a student is iteratively trained with pseudo-labels generated by the teacher. This may lead to noisy pseudo-labels and impact performance. To address this limitation, we introduce a new model called RE-AUM-LLM that generates high-quality pseudo-labels using self-training combined with Area Under the Margin (AUM) and Large Language Models (LLMs), such as Llama 3.1. Experimental results on two benchmark datasets show that the proposed approach achieves state-of-the-art results for low-resource relation extraction by comparison with several strong baselines. We will make the code publicly available to enable reproducibility and further research in this area.**

***Index Terms*—relation extraction, semi-supervised learning, low resource setting, self-training, area under the margin, LLM**

I. Introduction

Knowledge graphs typically represent structural knowledge from various domains in the form of triples, where each triple consists of a *head entity*, *relation*, and *tail entity* [1]. A knowledge graph relation extraction task identifies relevant relations between predefined head and tail entities. For example, from the sentence "The bird flew into the garden flower to find food," we can extract a relation *entity-destination* between the head entity "bird" and the tail entity "flower". Relation classification [2] is a sub-task of relation extraction that classifies relations into a predefined fixed set of categories given a text containing head and tail entities. Traditionally, relation extraction is done using supervised learning methods [3]–[5]. However, the main downside of supervised learning is that a large amount of labeled data is required for training.

Due to the labor-intensive and time-consuming process of labeling large datasets, prior works have used different forms of distant supervision to reduce human effort [6]. Such methods generally rely on a large knowledge base to extract relations and can introduce context-agnostic label noise, as the relation between entities may not reflect the current context. Moreover, semi-supervised methods for relation extraction, such as self-ensembling [7] and self-training [8], have been used to address the scarcity of labeled data. Semi-supervised learning approaches utilize a large amount of unlabeled data together with a small amount of labeled data. Self-ensembling methods aim to maintain stable predictions on unlabeled data despite model perturbations, but struggle with limited supervision, hindering the model's capacity to acquire new knowledge from unlabeled data and impeding further progress. Conversely, self-training methods aim to enhance the model by iteratively acquiring high-confidence labels from unlabeled data and using them to update the model. Nonetheless, directly employing self-training can lead to a gradual drift problem [9], [10], i.e., introducing noisy pseudo-labels that adversely affect the model performance.

[11] introduced a method that generates high-quality pseudo-labels from the unlabeled dataset and leverages a meta-learning technique during pseudo-label generation, with pseudo-label selection and exploitation, to mitigate the impact of noisy labels autonomously. However, the pseudo-label selection mechanism still uses a ratio of high-confidence pseudo-labeled instances and adds these instances directly to the labeled set for the next training iteration. Adding high-confidence pseudo-labeled instances does not necessarily improve the model significantly, as such instances offer limited additional information. Furthermore, those instances may also cause a gradual drift if the model is poorly calibrated. Other recent approaches, like [12] and [13], utilize contrastive learning in low-resource relation extraction. However, these approaches can increase model complexity and require manual supervision to extract task-specific templates.

In contrast to the prior approaches for relation extraction in low-resource settings, we aim to examine the impact of pseudo-label quality in self-training methods by observing the training dynamics of the pseudo-labels through AUM values. More specifically, instead of directly using the pseudo-label confidence of the teacher model, we utilize the AUM values [14] corresponding to the pseudo-labeled instances to harness differences in the training dynamics of clean and mislabeled/noisy instances. The AUM values are known to

show a strong correlation with the quality of the instances, with high AUM values indicating high-quality pseudo-labels and low AUM values indicating noisy pseudo-labels. AUM-ST [15] leverages training dynamics of unlabeled data to enhance self-training for text classification that is evaluated across diverse tasks, including emotion detection, sentiment analysis, and grammaticality classification on datasets from Reddit, Twitter, and online forums. However, we may still need to address noisy pseudo-labels. To handle noisy pseudo-labels, we additionally explore the robust generalization ability of LLMs and their effectiveness in relation extraction and use Meta's Llama3 model [16] as a data annotator for instances with low AUM values. Following an iterative self-training architecture, we show that the implementation of more refined pseudo-label quality assurance measures for pseudo-label selection combined with Llama pseudo-labels generated through in-context learning (ICL) [17], [18] could enhance the generalization performance of the model. To summarize, our contributions are as follows:

- We propose a new low-resource semi-supervised approach for relation extraction called RE-AUM-LLM.
- The proposed approach leverages AUM values that capture the training dynamics to identify noisy pseudo-labels. Noisy instances are relabeled by performing ICL with Llama 3.1.
- Experimental results on two benchmark datasets for relation extraction, SemEval, and TACRED, show that the proposed approach achieves state-of-the-art performance.

II. Proposed Approach

A. Proposed Model Architecture

Our proposed RE-AUM-LLM model for semi-supervised relation extraction is an adaptation of the AUM-ST [15] along with Incremental Meta Self-training (MetaSRE) architecture [11]. Similar to the MetaSRE, we have two main components in the teacher network, i.e., a relational statement encoder and a classification layer. In addition, our model incorporates an AUM network for assessing pseudo-label quality, similar to AUM-ST. Furthermore, our model uses an LLM to verify the lower-quality pseudo-labels. Algorithm 1 details the steps of the training process for RE-AUM-LLM, while a graphical representation of the overall architecture of the model is shown in Figure 1. Our changes compared to the original architectures in [11] and [14] are highlighted in blue color.

B. Teacher Network

Given a pair of head and tail entities, E_1 and E_2 in a sentence s, a relation statement (a.k.a., contextualized entity pair) is defined as a triple $x = (s, E_1, E_2)$. Our task is to train a model to map a relational statement x to a contextualized relation representation h and subsequently classify the relation statement x represented as h into one of several predefined relation categories. The role of the teacher network is to generate the relation statement representation h and perform the classification into specific relation categories. The teacher network consists of a relation statement encoder that uses BERT [19] to extract entity features. Following [20], [21], the relation statement provided as input to the encoder is represented as:

$$x = [w_1, \ldots, [E_{1_{\text{start}}}], w_i, \ldots, w_{j-1}, [E_{1_{\text{end}}}], \ldots, [E_{2_{\text{start}}}], w_k, \ldots, w_{l-1}, [E_{2_{\text{end}}}], \ldots, w_T]$$

A relation statement representation is obtained using entity-level instead of sentence-level output.

$$h = [h_{E1_{\text{start}}}, h_{E2_{\text{start}}}]$$

The teacher network consists of two main components: a relation statement encoder module responsible for converting the input x into contextualized representations h of entity pairs and a densely connected layer that utilizes h for classification. As standard in self-training, we use the teacher network, initially trained with labeled data $L = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, to generate pseudo-labels for the unlabeled data $U = \{\hat{x}_1, \ldots, \hat{x}_k\}$.

C. AUM-based Pseudo-Label Selection

In pseudo-label selection, conventional self-training methods rely on confidence scores to identify the most confident pseudo-labeled samples (U_b) and incorporate them into the training dataset for the subsequent iteration of the student network. The use of iterative self-training steps with the additional U_b data helps the model achieve high-quality pseudo-labels. However, only considering the confidence score for pseudo-label selection could contribute to a gradual drift problem. Therefore, in addition to the confidence score, we leverage AUM [14] as used in the Area Under the Margin Self-Training approach [15] to understand how much the pseudo-labels of the unlabeled data fluctuate during the training epochs. A high AUM score generally corresponds to high-quality pseudo-labels, while a low AUM corresponds to noisy pseudo-labels.

As suggested in the original AUM study [14], we use a set of intentionally mislabeled instances to determine an AUM threshold γ for the noisy data as the c^{th} percentile AUM value for the intentionally mislabeled instances. Subsequently, we split the unlabeled high-confidence data U_b into high AUM and low AUM pseudo-labels. High AUM pseudo-labels are directly added to the training set for the next self-training iteration, U_p, while low AUM pseudo-labels are first verified with an LLM and added to U_p only if they agree with the pseudo-labels generated by the LLM. We initially use a higher value for c to ensure that only good quality high AUM pseudo-labels from the teacher are included in U_p. However, selecting a high percentile value for threshold examples might result in easy-to-learn samples being erroneously categorized as low AUM, excluding valuable training data. To address this issue, we progressively decrease the percentile of threshold examples c selected in each iteration, which in turn results in a lower AUM threshold and the inclusion of more pseudo-labels from the teacher. Hence, AUM helps the model generalize better by identifying the potentially mislabeled data and evaluating the contribution of training examples. Furthermore, we use

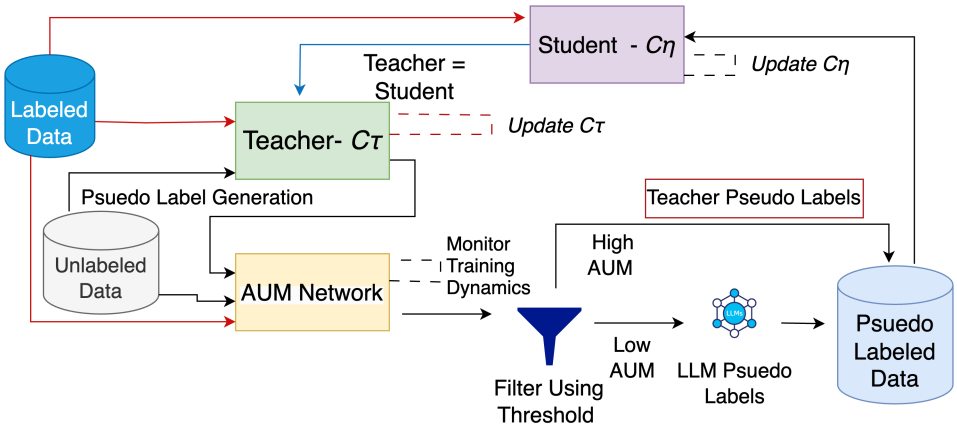

Fig. 1: RE-AUM-LLM Model Architecture - Semi-supervised relation extraction using ST with AUM and LLMs. Solid arrows represent the flow of data during the training phase, while dashed lines indicate loss computation. The diagram consists of three main components: Student Network, Teacher Network, and AUM Network. Teacher Network is trained with x% of labeled data from the training set. For the y% unlabeled data, pseudo-labels are generated using the teacher network. An AUM Network is then initialized to monitor the training dynamics of the pseudo-labels. Using intentionally mislabeled threshold examples, the AUM Network filters the pseudo-labels into High AUM and Low AUM sets. For High AUM, the teacher network pseudo-labels are used to train the student network, whereas the Low AUM pseudo-labels are verified using LLM and added to the train set. A student network is thus trained using labeled data along with (high-confidence) high-AUM and LLM-verified low-AUM data. In the subsequent iteration of self-training, the student network adopts the teacher's parameters as its starting point, and the training process continues from there.

the generalization capability of LLMs to verify the low-AUM pseudo-labels generated by the generation network with LLM-generated labels to produce high confidence, high-average margin, and LLM-verified pseudo-labeled samples.

Intuitively, high confidence, high AUM unlabeled instances are considered to have better quality pseudo-labels. However, since we initially train the teacher network with a limited amount of labeled data, a stricter/higher AUM threshold needs to be used to filter the good-quality data. As the training progresses and the model improves, the AUM threshold is decreased to include potentially valuable instances with lower AUM values. To verify low-quality AUM pseudo-labels, we use in-context learning to identify relations between entities in a given sentence using a similar approach to [22]. A prompt consisting of *Task Description*, *Demonstration Examples*, and *Test Input* is constructed for each test example and fed to the LLM model (specifically, Llama3.1). We use SIMCSE [23] for computing the similarities between relation statement representations, specifically between the representation of the test statements and the representations of the training statements, as shown in Figure 2.

D. Student Network

After selecting (high-confidence) high-AUM and LLM-verified low-AUM samples from pseudo-labeled data generated by Teacher Network, we use this data, denoted by U_p combined with the labeled data L to (re-)train the Student Network. The Student Network has a similar architecture to the Teacher Network but is initialized and trained independently. After obtaining the high-quality filtered pseudo-labels using the recently updated Teacher Network and AUM Network, we train the student network using the labeled data and the filtered pseudo-labeled data from the unlabeled set. After one iteration of training, this network is used as the Teacher Network to generate pseudo-labels, and the training continues for N self-training steps.

III. Experimental Setup

In this section, we describe the datasets used, baseline approaches, and evaluation metrics. Other details of the experimental setup, including hyperparameters used for the proposed approach, are presented in Section III-E.

A. Datasets

We used two benchmark datasets, SemEval 2010 Task 8 [24] and TACRED [25], which are commonly used to evaluate relation extraction approaches.

SemEval. SemEval 2010 Task 8 [24] is a publicly available benchmark dataset collected from general domain resources. It captures semantic relations between pairs of entities that represent nominals (i.e., nouns and noun phrases). The entities, which have been predefined and marked in the dataset, are linked through one of 19 relation categories, including the "no_relation" category, which accounts for 17.4% of the data.

TACRED. The TAC Relation Extraction Dataset (TACRED) [25] is a large-scale collection designed for relation extraction tasks. Each sentence contains a pair of head and tail entities linked by one of 42 relation types, including "no_relation," which accounts for 78.7% of the data.The tail and head entities have been predefined and marked in the dataset. Compared to SemEval, TACRED is more challenging due to its greater variety of relation types and a more heavily skewed distribution of negative mentions.

Following [10] and other works, we divided the training set into labeled and unlabeled subsets using stratified sampling.

Algorithm 1 RE-AUM-LLM Model Training

Require: Labeled data, $L = \{(x_1, y_1), \ldots, (x_n, y_n)\}$; Unlabeled data, $U = \{\hat{x}_1, \ldots, \hat{x}_k\}$; the confidence threshold Z used to select pseudo-labels for the next iteration; the percentile c of mislabeled data used to determine the AUM threshold.

1) Train the Teacher Network C_τ using labeled data (L) by minimizing the cross entropy loss:

   $L_{C_\tau} = \sum_{i=1}^{n} \text{loss}(C_\tau(x_{n,E_1,E_2}), \text{one\_hot}(y_n))$, where E_1 and E_2 are given head and tail entities in instance x_n

2) For **k** Self-Training steps, do:
   a) Using updated Teacher Network C_τ, generate pseudo-labels for unlabeled data $U = \{(\hat{x}_1, \hat{y}_1), \ldots, (\hat{x}_k, \hat{y}_k)\}$
   b) Filter the generated pseudo-labels using the confidence threshold Z and construct $U_b = \{(\hat{x}_1, \hat{y}_1), \ldots, (\hat{x}_k, \hat{y}_m)\}$
   c) Select Threshold_Samples as a subset of U_b (e.g., for N samples and cl classes, select N/(cl + 1)) and assign to them a label corresponding to a new, non-existent class
   d) Train a C_{AUM} model on both L and U_b and monitor the training dynamics of U_b instances over **t** epochs. Calculate the AUM score for a pseudo-labeled instance $(\hat{x}_i, \hat{y}_i)$ by averaging its margin across the epochs:

      $\text{AUM}(\hat{x}_i, \hat{y}_i) = \frac{1}{t} \sum_{i}^{t} \left[z_{\hat{y}_i} - \max_{\hat{y}_i \neq j}(z_j) \right]$

      where $z_{\hat{y}_i}$ is the logit corresponding to $\hat{y}_i$ and z_j is the logit corresponding to the largest other logit.
   e) Calculate the AUM threshold to be used to identify noisy pseudo-labels as the c^{th} percentile AUM value:

      $\gamma \leftarrow$ Mislabeled_Samples_AUM_values[(len(Mislabeled_Samples_AUM_values)) $\times$ c]

   f) Rank and select the high and low AUM examples using the AUM threshold γ:

      $U_{HIGH} = \{(\hat{x}_i, \hat{y}_i) \in U_b \mid \text{AUM}(\hat{x}_i, \hat{y}_i)$¿$\gamma\}$ and $U_{LOW} = \{(\hat{x}_i, \hat{y}_i) \in U_b \mid \text{AUM}(\hat{x}_i, \hat{y}_i)$¡$= \gamma\}$

   g) Set $U_{PL} = U_{HIGH}$
   h) Use an LLM model to generate pseudo-labels for noisy instances, i.e., U_{LOW}. Add each $(\hat{x}_i, \hat{y}_i) \in U_{LOW}$ to U_{PL} if the pseudo-label generated by LLM equals the label generated by Teacher Network C_τ
   i) Now, $U_{PL} = \{(\hat{x}_i, \hat{y}_i), \ldots, (\hat{x}_p, \hat{y}_p)\}$
   j) Finally, train the Student Network C_η using labeled data with selected high confidence, high-average-margin and low-average-margin LLM-verified pseudo-labeled data.

      $L_{C_\eta} = \sum_{i=1}^{n} \text{loss}(l_n, \text{one\_hot}(y_n)) + \sum_{i=1}^{p} \text{loss}(l_p, \text{one\_hot}(y_p))$

   k) Decrease the percentile c used to calculate the AUM threshold γ (which will results in a lower threshold γ)
   l) Use the Student Network as teacher $C_\tau = C_\eta$ and go back to Step 2

The percentages cover very low (3-5%), moderate (10%), and high (15-30%) amounts of labeled data, allowing for a comprehensive evaluation of model performance across different resource availability scenarios. TACRED is approximately 10 times the size of SemEval. Therefore, such splits were generated to create comparable low-resource scenarios across both datasets despite their size differences. Specifically, subsets consisting of 5%, 10%, and 30% as labeled data and 50% as unlabeled data are generated for SemEval. Similarly, subsets consisting of 3%, 10%, and 15% as labeled data and 50% as unlabeled data are generated for TACRED. Each subset was generated 5 times using 5 different random seeds. Statistics of the datasets, including Train, Dev, and Test size, as well as the size of the subsets generated, are shown in Table I.

B. Baselines

Our RE-AUM-LLM model is compared against supervised BERT [19] baselines, and a variety of semi-supervised low-resource baselines, as described below. Since BERT was the best-performing model identified in [11], it is used as the base encoder for all semi-supervised learning approaches.

BERT. The BERT [19] model was trained only with the small percentage of labeled data available in each experiment in a supervised setting. As such, it can be seen as a lower bound baseline for the semi-supervised approaches.

BERT with gold labels. We also utilize a supervised BERT model trained on a small percentage of label data combined with all data without holding out labels (unlabeled), as it is standard in related works. This model can be seen as providing an upper-bound baseline for the semi-supervised approaches.

Self-Training (ST). In the basic Self-Training approach, the teacher network provides pseudo-labels, which the student network then utilizes for improving the predictions [26].

Mean-Teacher (MT). In Mean-Teacher, similar augmented/perturbed inputs are provided to the teacher and student models. These models aim to make similar predictions by minimizing classification and consistency under perturbation losses [27]. The teacher weights are obtained as an Exponential Moving Average(EMA) of the student weights.

DualRE-Pointwise & DualRE-Pairwise. The DualRE framework [28] introduces a dual relation statement retrieval module trained together with the original relation prediction module. The statement retrieval module is trained using a Pairwise or Pointwise learning-to-rank approach.

RE-Ensemble. The RE-Ensemble [28] is similar to the DualRE framework, but it replaces the retrieval component in DualRE with a second identical prediction module (with different weights), creating an ensemble of predictions from two similar models to improve accuracy.

MRefG. In the Multiple Reference Graphs (MRefG) approach [29], the author constructs reference graphs using entity, verb, and semantic references to connect unlabeled instances to labeled instances, aiming for semantic or lexical connection, and employs the MRefG model to utilize this reference information

Dataset	#Relations	#Train	#Dev	#Test	#Unlab	#3%	#5%	#10%	#15%	#30%
SemEval	19	7,199	800	1,864	3,599	-	559	719	-	2,279
TACRED	42	75,049	25,763	18,659	37,524	2,251	-	7,504	11,257	-

TABLE I: Summary of SemEval and TACRED datasets. 50% of the train set is used as unlabeled data, and x% of the train set (i.e., 5, 10, 30% for SemEval, and 3, 10, 15% for the TACRED dataset) is used as labeled data for training.

for improved recognition of high-quality unlabeled data.

MetaSRE. Semi-supervised Relation Extraction via Incremental Meta Self-Training (MetaSRE) [11] incrementally trains relation classification and relation label generation networks using a meta-learning objective with a pseudo-label selection and exploitation scheme to assess the quality of pseudo-labeled data generated.

GradLRE. The Gradient Imitation Reinforcement Learning for Low-resource Relation Extraction (GradLRE) model [30] encourages pseudo-labeled data to align with the gradient optimization path of labeled data, with the goal of improving the pseudo-label quality.

SelfLRE. The Self-refining Representation Learning for Low-resource Relation Extraction (SelfLRE) model [12] integrates two complementary approaches, self-training and self-ensembling, and is jointly trained through multi-task learning to improve the extraction of relations.

UG-MCT. The Uncertainty-Guided Mutual Consistency Training framework (UG-MCT) [31] utilizes two models with identical architectures but different weights, together with a pseudo-labeling module, and leverages uncertainty to guide the model in prioritizing confident pseudo-labels while reducing noise from inaccurate ones.

TempCL. TempCL [13] introduces a template-based method that reduces noise by focusing on meaningful entities and the semantic information of their relationship. Instance-wise and group-wise contrastive learning aligns templates with original sentences in the same label group and separates those from different groups.

C. Evaluation Metrics

Our models predict the category y of a relation statement $x = (s, E_1, E_2)$, given the sentence s and the entities E_1 and E_2. Similar to related works, each model is run five times using subsets generated with different random seeds using stratified sampling. The results reported represent F1 scores averaged over the five runs. The results of the baselines are obtained directly from the respective works that published those models.

D. Prompt design

We have developed a prompt to facilitate pseudo-label generation using both 0-shot and k-shot learning approaches. In our experiments, we used $k = 15$ to generate 15 examples using task-aware demonstration retrieval [22] together with SimCSE [23], as shown in Figure 2. The prompt structure is divided into four components: a context, a header, an intermediate section, and a footer, as detailed below. While the context, header, and footer remain consistent across datasets, the intermediate section varies based on the dataset, reflecting the distinct structure of labels in each dataset. This design ensures that the prompt instructs the model to produce output in a specific format aligned with the dataset's label structure.

For example, in the SemEval dataset, relation labels follow the structure Relation(Entity1, Entity2). The segment before the parentheses defines the relationship type (e.g., "ownership," "membership," or "location"), while the terms within the parentheses represent the entities involved in the relationship. The order of these entities is critical as it determines their roles in the relationship. For instance, in the label *Component-Whole(e2, e1)*, "wheel (e2) is part of a car (e1)," whereas in *Component-Whole(e1, e2)*, "a car (e1) is part of an assembly line (e2)." This distinction shows the importance of maintaining the correct entity order for accurate relationship representation.

In the TACRED dataset, the term preceding the colon (e.g., org or per) specifies the category of the entity, while the term following the colon (e.g., founded, subsidiaries, date_of_birth, cause_of_death) defines the nature of the relationship or attribute associated with the entities. For instance, the label *org:founded* represents a founding relationship, where e1 corresponds to the organization and e2 denotes the founding date. Similarly, *per:date_of_birth* identifies the birth date of an individual, where e1 refers to the person, and e2 indicates the date. The distinction between the 0-shot and 15-shot approaches lies in the footer section. In the 15-shot setting, a list of 15 examples is included in the footer, while all other components remain identical between the two configurations.

- Context
  - *You are an expert in natural language processing, specializing in understanding relationships between entities in text. Your task is to identify and output the relationship between two entities enclosed within the tags* <e1> *and* <e2>.
- Header
  - *Analyze the given test sentence carefully and determine how the entities inside the tags* e1 *and* e2 *are related. Please output the relationship between entities in the format below:* <relation></relation>. *Select only one option from the predefined relations list given in the list below:* {predefined_relations}
- Intermediate
  - *SemEVAL: The relationship in the list denotes the flow of relation from entity e1 to e2 or e2 to e1.*
  - *TACRED: The relationship in the list denote how entity e1 is related to e2 separated by a colon(:).*
- Footer

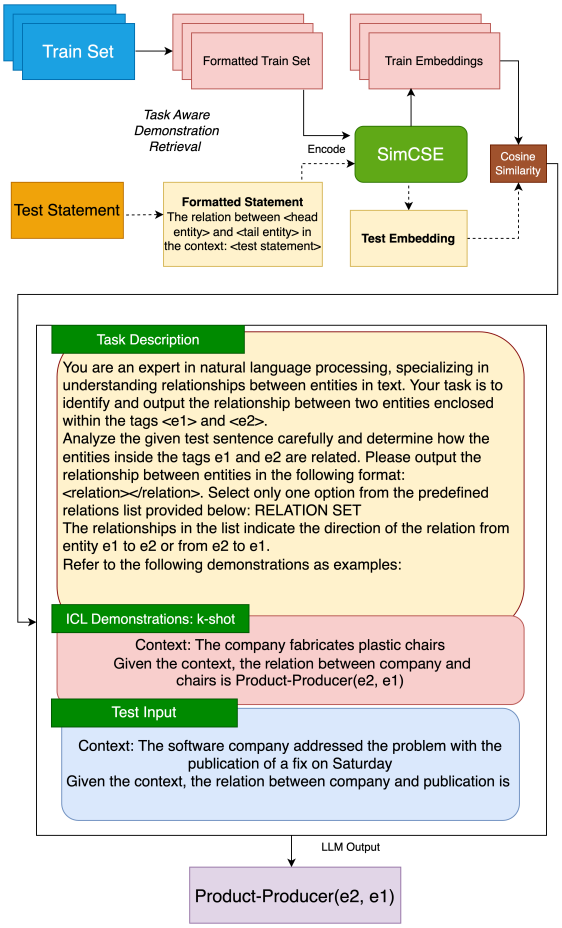

Fig. 2: Illustration for prompt construction for task-aware demonstration retrieval. Given the test input, we construct a prompt with a task description and ICL Demonstration samples. Samples are embedded using task-aware demonstration retrieval [22] together with SimCSE [23]. This process ensures higher-quality samples similar to the test input.

- Only in 15-shot: *Use the following demonstrations as examples:* {`list of 15 examples`}.
- *Test Sentence:* {`original_text`} *is:*

E. Implementation Details

The contextualized relational encoder modules in the Teacher, Student, and AUM Networks are initialized with the *BERT-Base+Cased* model weights with the default BERT tokenizer and max-length 128. Loss optimization is done using BERTAdam. We utilize a fully connected network having layer dimensions $2 \cdot h_R - h_R -$ label_size, where $h_R = 768$. The learning rate is set to 5e-5, warm-up is 0.1, the number of self-training steps is 10 (k =10, Algorithm 1 Step 2), the number of supervised training epochs is 10 (Algorithm 1 Step 1), the number of AUM training epoch is 4 (Algorithm 1 Step 2(d)) and the number of self-training epoch is 15 (Algorithm 1 Step 2(j)). The supervised training batch size is 16 (Algorithm 1 Step 1), and the self-training batch size is 32 (Algorithm 1 Step 2(j)). The confidence threshold is set to Z = 70% (Algorithm 1 Step 2(b)), and percentile c is set to 0.95. For in-context learning with demonstration selection using Llama3.1, the hyperparameters used are engine='llama3.1-70b-instruct', temp=0.3, and max-tokens=200. Training was conducted on an NVIDIA A40 GPU with 46068 MiB of memory.

IV. Experimental Results

We first present experiments with three LLMs in zero and in-context k-shot settings, performed to identify a good LLM to be used with the RE-AUM-LLM approach. Next, we present the results of our model by comparison with baseline results, and finally, we perform an ablation study.

A. LLM Results

To identify a good LLM to be used with the RE-AUM-LLM approach, in this section, we present the results of in-context learning with k-shot demonstration selection using ChatGPT, Llama3.1-70b, and Qwen2-72b in Table II. Specifically, the models were evaluated in zero-shot and 15-shot settings. The 15-shot setting was selected following [22]. The results show that LLama3.1-70b and Qwen2-72b have similar performance in both settings for both datasets. Qwen2-72b achieves the best F1-score of 58.99%, followed by LLama3.1-70b with an F1-score of 58.33% on SemEval in the 15-shot setting. Llama3.1-70b achieves the best F1-score of 24.92%, followed by Qwen2-72b with an F1-score of 23.27% on TACRED in the 15-shot setting. Since Llama3.1-70b has the overall best performance, we use this model to acquire pseudo-labels for the noisy low-AUM instances in our semi-supervised experiments.

Model	K-shot	SemEval	TACRED
ChatGPT	0	26.32	15.40
	15	38.90	24.81
Llama3.1-70b	0	**34.54**	16.67
	15	58.33	**24.92**
Qwen2-72b	0	28.26	**17.41**
	15	**58.99**	23.27

TABLE II: LLM comparison in terms of F1 on the test set, in zero-shot and 15-shot settings. The best results for each dataset in each setting are highlighted in **bold**.

B. RE-AUM-LLM Results

The results of the RE-AUM-LLM approach on the SemEval and TACRED datasets by comparison with the results of the baselines considered are presented in Table III for varying percentages of labeled data. Notably, the RE-AUM-LLM model has the best overall performance compared to prior models, with better performance than the best prior model, TempCL, in 5 out of 6 cases. More specifically, our model exhibits better performance than the TempCL for lower 5% and 10% resource settings in the case of the SemEval dataset and in all low-resource settings for the TACRED dataset. For the 30% setting of the SemEval dataset, the RE-AUM-LLM model has an F1-score of 86.63, which is very close to 86.93 score of the TempCL model. This is a remarkable result, given that the

	SemEval			TACRED		
Methods	**5%**	**10%**	**30%**	**3%**	**10%**	**15%**
BERT [19]	$70.71_{\pm 1.24}$	$71.93_{\pm 0.99}$	$78.55_{\pm 0.87}$	$40.11_{\pm 3.88}$	$53.17_{\pm 1.67}$	$55.55_{\pm 0.82}$
ST [26]	$71.34_{\pm 1.68}$	$74.25_{\pm 1.10}$	$81.71_{\pm 0.79}$	$42.11_{\pm 1.04}$	$54.17_{\pm 0.53}$	$56.52_{\pm 0.40}$
MT [27]	$70.05_{\pm 3.89}$	$73.37_{\pm 1.42}$	$80.61_{\pm 0.81}$	$44.34_{\pm 1.78}$	$53.08_{\pm 1.01}$	$53.79_{\pm 1.38}$
RE-Ensemble [28]	$72.35_{\pm 2.63}$	$75.71_{\pm 1.39}$	$81.34_{\pm 0.74}$	$42.78_{\pm 1.89}$	$54.83_{\pm 0.95}$	$55.68_{\pm 1.21}$
DualRE-Pairwise [28]	$74.35_{\pm 1.76}$	$77.13_{\pm 1.10}$	$82.88_{\pm 0.67}$	$43.06_{\pm 1.73}$	$56.03_{\pm 0.55}$	$57.99_{\pm 0.67}$
DualRE-Pointwise [28]	$74.02_{\pm 1.68}$	$77.11_{\pm 1.02}$	$82.91_{\pm 0.62}$	$43.73_{\pm 1.60}$	$56.28_{\pm 0.61}$	$57.72_{\pm 0.49}$
MRefG [29]	$75.48_{\pm 1.34}$	$77.96_{\pm 0.90}$	$83.24_{\pm 0.71}$	$43.81_{\pm 1.44}$	$55.42_{\pm 1.40}$	$58.21_{\pm 0.71}$
MetaSRE [11]	$78.33_{\pm 0.92}$	$80.09_{\pm 0.78}$	$84.81_{\pm 0.44}$	$46.16_{\pm 1.02}$	$56.95_{\pm 0.34}$	$58.94_{\pm 0.36}$
GradLRE [30]	$79.65_{\pm 0.68}$	$81.69_{\pm 0.57}$	$85.52_{\pm 0.34}$	$47.37_{\pm 0.74}$	$58.20_{\pm 0.33}$	$59.93_{\pm 0.31}$
UG-MCT [31]	$80.43_{\pm 0.52}$	$82.91_{\pm 0.43}$	$85.99_{\pm 0.31}$	$45.10_{\pm 1.36}$	$57.97_{\pm 0.41}$	$61.33_{\pm 0.28}$
SelfLRE [12]	$81.24_{\pm 0.53}$	$83.42_{\pm 0.49}$	$86.35_{\pm 0.47}$	$51.16_{\pm 1.39}$	$60.06_{\pm 1.44}$	$62.39_{\pm 0.41}$
TempCL [13]	$82.14_{\pm 0.64}$	$83.97_{\pm 0.20}$	$\mathbf{86.93}_{\pm 0.28}$	$52.89_{\pm 0.29}$	$60.45_{\pm 0.24}$	$62.75_{\pm 0.31}$
RE-AUM-LLM (Ours)	$\mathbf{82.19}_{\pm 0.54}$	$\mathbf{84.05}_{\pm 0.39}$	$86.63_{\pm 0.30}$	$\mathbf{52.96}_{\pm 0.55}$	$\mathbf{60.55}_{\pm 0.41}$	$\mathbf{62.78}_{\pm 0.82}$
BERT with gold labels	$84.64_{\pm 0.28}$	$85.40_{\pm 0.34}$	$87.08_{\pm 0.23}$	$62.93_{\pm 0.41}$	$63.66_{\pm 0.23}$	$64.69_{\pm 0.29}$

TABLE III: Performance comparison across models with varying percentages of labeled data and 50% unlabeled data on the SemEval and TACRED datasets. Results are reported in terms of F1-scores averaged over five runs (together with standard deviation values). The best results for each dataset in each setting are highlighted in **bold**

templates in TempCL are manually designed and do not easily generalize to domains for which templates are not available without additional manual effort.

It is also interesting to note that the results of our model are very close to the results of the supervised BERT model that uses all the unlabeled data as gold labels (in addition to the respective percentages of labeled data). For example, for the 30% setting of SemEval, the supervised BERT with gold labels has a score of 87.08, just slightly higher than the score of 86.63 of our model. Overall, the gap between the performance of the two models decreases as the percentage of labeled data increases.

C. Ablation Study

As part of ablation studies, we conducted experiments to study the contribution of the low AUM pseudo-labels, as well as the contribution of the LLM verification component to the overall performance of our proposed model. Specifically, we compare RE-AUM-LLM with a variant (*w/o Low AUM*) where the low AUM pseudo-labels are not included in the semi-supervised model training, and also with a variant (*w/o LLM Verification*) where the low AUM pseudo-labels are all included in the training without being verified with the LLM. The results are presented in Table IV. As can be seen, both components contribute to the performance of the model, as both variants lead to a decrease in performance. However, between the two components, the exclusion of the low AUM pseudo-labels (*w/o Low AUM*) seems to affect more the performance for the larger TACRED dataset. For SemEval, (*w/o Low AUM*) affects the performance more in the 5% labeled data setting, while including the low AUM pseudo-labels without verifying them (*w/o LLM Verification*) affects the performance more for the 10% and 30% settings. The importance of one component relative to the other seems to be related to the amount of labeled data used. If the amount of labeled data is small, including and verifying the low-AUM pseudo-label is important for increasing the amount of training data without introducing noise. If the amount of labeled data is larger, the low AUM pseudo-labels are more accurate, and verification with LLMs may be less critical.

D. RE-AUM-LLM versus TempCL/SelfLRE

Baseline models such as TempCL [13] and SelfLRE [12] require careful manual design of templates/relation descriptions. They present the idea of manually creating a template corresponding to each relation type, which is a crucial step in these approaches. The template may be ambiguous depending on how they were created in the first place, and in domain-specific scenarios could cause an issue as one needs to understand not just the sentence-level relationships between entities but also the document-level relationships between entities. As opposed to that, we propose to leverage the generalization capabilities of LLMs as label verifiers in combination with AUM for pseudo-label selection, an approach that has not yet been studied in the context of semi-supervised low-resource relation extraction.

As expected, predicting the unlabeled data using an LLM adds some computational cost, although this can be done once for the whole unlabeled dataset, independent of the self-training iterations. Furthermore, the AUM training adds some cost to the overall training process. However, the overall time is still dominated by the self-training iterations, as can be seen in Section IV-F. A direct time comparison between the RE-AUM-LLM and the TempCL/SelfLRE baselines is not feasible, although it is expected that these approaches also add some computational overhead to the self-training steps.

E. Importance of High AUM Examples

High AUM values generally help us to identify correctly labeled data. The AUM statistic gives the average separation between the logit of the correct class and the logits of the other classes for a given sample. In our work, we compare the logit of the pseudo-label to the logits of the other classes. High-AUM pseudo-labeled samples have a high probability of being correct, tending to exhibit a larger margin between the correct class and the incorrect classes. A weaker separation between the predicted class and other high-probability classes

	SemEval			TACRED		
Methods	**5%**	**10%**	**30%**	**3%**	**10%**	**15%**
RE-AUM-LLM	**82.19**$_{\pm 0.54}$	**84.05**$_{\pm 0.39}$	**86.63**$_{\pm 0.30}$	**52.96**$_{\pm 0.55}$	**60.55**$_{\pm 0.41}$	**62.78**$_{\pm 0.82}$
w/o Low AUM	78.44$_{\pm 0.69}$	82.57$_{\pm 0.57}$	85.64$_{\pm 0.72}$	48.76$_{\pm 0.73}$	58.68$_{\pm 1.81}$	60.94$_{\pm 1.5}$
w/o LLM Verification	80.79$_{\pm 0.62}$	81.73$_{\pm 0.87}$	85.03$_{\pm 1.34}$	49.21$_{\pm 1.23}$	59.64$_{\pm 0.71}$	61.44$_{\pm 0.84}$

TABLE IV: Ablation study for comparison of each component for RE-AUM-LLM on SemEval and TACRED

(smaller/negative values) indicates that the data is potentially mislabeled. Thus, High-AUM samples can be directly used as training data in subsequent self-training iterations.

The following examples demonstrate the difference between how we use High-AUM versus Low-AUM pseudo-labels.

High-AUM Example1 (TACRED): *There's been some very positive developments in recent weeks, in recent months, said <e1> Arsenal </e1> 's chief executive <e2> Ivan Gazidis </e2>, adding: "We 're seeing sales progressing well, and the development is on a sound financial footing."*

Ground Truth Relation Class: org:top_members/employees, AUM Value: 6.5112, Pseudo-Labeled Relation Class: org:top_members/employees.

High AUM Example2 (SemEval): *They have placed <e1>corpses</e1> into shallow <e2>graves</e2> along with stone tools*

Ground Truth Relation Class: Cause-Effect(e2,e1), AUM Value: 4.620, Pseudo-Labeled Relation Class: Cause-Effect(e2,e1).

High AUM Example3 (SemEval): *The <e1>band</e1> performs with a high level of <e2>musicality</e2>, energy and spirit while combining sensitive group interplay with dynamic solo improvisations*

Ground Truth Relation Class: Other, AUM Value: 2.566, Pseudo-Labeled Relation Class: Other.

Low AUM Example1 (TACRED): *<e2> Doherty </e2>, frontman for <e1> Babyshambles </e1> and formerly co-leader of The Libertines, gets more attention in Britain as the boyfriend of model Kate Moss*

Ground Truth Relation Class: no_relation, AUM Value: -14.29, Pseudo-Labeled Relation Class: per:spouse.

Low AUM Example2 (SemEval): *The following is a <e1>list</e1> showing the 100 largest incorporated <e2>cities</e2> in the state of California ranked by population*

Ground Truth Relation Class: Instrument-Agency(e1,e2), AUM Value: -8.31, Pseudo-Labeled Relation Class: Message-Topic(e1,e2).

Low AUM Example3 (SemEval): *He dragged the nail on the head of the snake and then clicked the <e1>switch</e1> of the <e2>tape recorder</e2>*

Ground Truth Relation Class: Component-Whole(e1,e2), AUM Value: -3.25, Pseudo-Labeled Relation Class: Product-Producer(e2,e1).

F. Computational Analysis

This section provides information about the time and resources required for training the proposed model. Our approach was trained on an NVIDIA A40 GPU and consumed about 17.17 GB of GPU memory. The total training time for each experiment for the SemEval Dataset with 5% labeled data and 50% unlabeled data is 1.14 hours. Table V summarizes the training time required for each component of the proposed method. Similarly, the total time required to train the model with 10% labeled and 30% labeled SemEval data and 50% unlabeled data was 1.18 hours and 1.61 hours, respectively.

V. Related Work

In this section, we discuss relevant semi-supervised relation extraction approaches for relation extraction, as well as relevant LLMs works.

A. Semi-Supervised Relation Extraction

Traditionally, relation extraction has been carried out in a supervised setting where the model classifies a relation statement into one of several relations categories [32]–[34]. The downside of this approach is that a large amount of data is required for training. Leveraging unlabeled data together with small labeled data in a semi-supervised setting represents a promising alternative. Semi-supervised methods iteratively improve the model performance by generating pseudo-labels on the unlabeled data and retraining the model on the pseudo-labels. There are two major categories of semi-supervised learning methods,i.e., self-training and self-ensembling methods. [8] proposed a framework where pseudo-labels are incrementally generated for unlabeled data to enhance the model's prediction. However, the pseudo-labels generated by the model can be noisy, which causes semantic drift [9], [10], [35], [36]. To overcome the shortcomings of other techniques [37] and avoid semantic drift, [11] proposed an incremental self-training mechanism with meta-learner. However, the pseudo-label selection filters out pseudo-labels based only on the confidence score and may still introduce noise in the initial iteration. A recent study [12] introduces SelfLRE, which combines self-training and self-ensembling methods. Pseudo-labels from self-training guide the refinement of task-specific representations, while the proximity of semantic representations with the same pseudo-labels helps correct pseudo-label errors. Another recent study [13], applies contrastive learning to low-resource relation extraction but requires manual effort to extract task-specific templates.

Similar to such works, we focus on semi-supervised relation extraction in a low-resource setting. Specifically, we adopt a similar architecture to ST but use AUM [14] to identify noisy pseudo-labels. [15] proposed a self-training approach that uses AUM to improve model performance by monitoring the training dynamics of unlabeled data.

Components	Training Time (Seconds)	Comments
Supervised Training	38.15	Teacher network training using the labeled dataset. Skipped in later training iterations.
Pseudo-labeling	3.97	Pseudo-labels generation using the Teacher Network.
AUM Training	55.80	AUM Network training using labeled and pseudo-labeled datasets.
Self-Training	235.88	Student Network training using all filtered pseudo-labeled datasets.

TABLE V: Computational time analysis of RE-AUM-LLM for one self-training iteration for SemEval 5% Labeled and 50% unlabeled dataset

B. Large Language Models

Utilizing the Transformer-based framework and building on the achievements of pre-trained language models like BERT across numerous NLP tasks, sophisticated models with impressive generalization abilities, including OpenAI's GPT (Generative Pre-Training Transformer) series [38], Meta's LLaMA3 (Large Language Model Meta AI) [16], Qwen2 [39], etc. have been introduced. Numerous research using such LLMs have shown notable outcomes in downstream tasks with zero and few demonstration examples with in-context learning [40], [41]. From a practical point of view, it is not feasible to fine-tune LLMs for downstream tasks; therefore, using in-context learning techniques, the models can learn through examples/instruction prompts. Formulating effective prompts for downstream tasks entails employing different techniques in prompt engineering, such as the Chain of Thought prompting (COT) [42], Demonstration Selection (DS) [43], and Self-consistency (SC) [44], among others.

Recent works on relation extraction have explored the use of LLMs and the effectiveness of different prompt engineering techniques in terms of generalization to domain-specific and general-domain tasks. For example, [22] shows the effectiveness of GPT-3 in both general and scientific domains. Other works [45]–[47] have also shown that dynamic demonstrations selected for each test example show improved results. Selecting demonstration examples that are nearest neighbors to the test example generally leads to better results. Sentence embeddings are used for retrieval, such as Sentence-BERT [48], SimCSE [23]. However, using sentence embeddings to retrieve k-nearest neighbors for demonstration is not effective for the relation extraction task [22]. To address this limitation, [22] proposed a task-aware demonstration retrieval method, which specifically emphasizes the entities and relation information in the embedding, as opposed to simply embedding the sentence. We use this approach together with SimCSE [23] to represent statements and identify informative demonstration examples for relation extraction. *To the best of our knowledge, RE-AUM-LLM is the first approach that combines ST, AUM, and LLMs for relation extraction in a low-resource setting.*

VI. Conclusions and Future Work

This paper proposes a semi-supervised learning framework for self-training informed by AUM and LLMs as an external verifier of the low-AUM noisy pseudo-labels. The framework addresses the gradual drift problem during model training by reducing noises in the pseudo-label generation. Instead of selecting only pseudo-labels with high confidence, we use AUM to filter those pseudo-labels that either have high-AUM, or have low-AUM but can be verified with an LLM. Experiments on two benchmark datasets, SemEval and TACRED, show improvements over baselines, the effectiveness of using AUM values, and the potential of LLMs for reducing noisy pseudo-labels. A future direction of this research is to apply similar semi-supervised learning techniques in other information extraction tasks, such as triple extraction.

VII. Limitations

This work has some limitations that we need to acknowledge. First, we assume that the entities are provided, and we identify the type of relation between them. However, in practical applications, the entities would also need to be extracted. Thus, it would be of interest to design semi-supervised low-resource approaches that identify not only the relation category, but the whole triple consisting of entities and the relation between them. As another limitation, the relation identification in this work is at the sentence level. Extracting relations at the document level, by integrating information within a sentence and across multiple sentences of a document, in a low-resource setting, cannot be performed with the current approach. In terms of LLMs, we should note that the datasets that are used to pre-train the LLMs are unknown, but overall may capture a significant amount of knowledge in the general domain. The capabilities of the models in more specific domains are not very well understood, and it is not clear if the LLMs would show similar performance on relation extraction in more specific domains. This highlights the need for domain-specific benchmark datasets for relation extraction in low-resource settings.

VIII. Acknowledgment

This research was sponsored by the Department of the Navy, Office of Naval Research under ONR award number N00014-21-1-2286, and also by the Cognitive and Neurobiological Approaches to Plasticity (CNAP) Center of Biomedical Research Excellence (COBRE) of the National Institutes of Health (NIH) under grant number P20GM113109. The content is solely the responsibility of the authors and does not necessarily represent the official views of ONR/NIH.

REFERENCES

[1] N. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor, "Industry-scale knowledge graphs: Lessons and challenges: Five diverse technology companies show how it's done," *Queue*, vol. 17, no. 2, pp. 48–75, 2019.

[2] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *SDM06: workshop on link analysis, counter-terrorism and security*, vol. 30, 2006, pp. 798–805.

[3] W. Zeng, Y. Lin, Z. Liu, and M. Sun, "Incorporating relation paths in neural relation extraction," *arXiv preprint arXiv:1609.07479*, 2016.

[4] Y. Zhang, V. Zhong, D. Chen, G. Angeli, and C. D. Manning, "Position-aware attention and supervised data improve slot filling," in *Conference on empirical methods in natural language processing*, 2017.

[5] J. Y. Huang, B. Li, J. Xu, and M. Chen, "Unified semantic typing with meaningful label inference," *arXiv preprint arXiv:2205.01826*, 2022.

[6] A. Smirnova and P. Cudré-Mauroux, "Relation extraction using distant supervision: A survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 5, pp. 1–35, 2018.

[7] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.

[8] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-supervised self-training of object detection models," 2005.

[9] S. Liu, A. Davison, and E. Johns, "Self-supervised generalisation with meta auxiliary learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[10] W. Li and T. Qian, "Exploit multiple reference graphs for semi-supervised relation extraction," *arXiv preprint arXiv:2010.11383*, 2020.

[11] X. Hu, C. Zhang, F. Ma, C. Liu, L. Wen, and P. S. Yu, "Semi-supervised relation extraction via incremental meta self-training," *arXiv preprint arXiv:2010.16410*, 2020.

[12] X. Hu, J. Chen, S. Meng, L. Wen, and P. S. Yu, "Selflre: Self-refining representation learning for low-resource relation extraction," in *Proceedings of the 46th Int. ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023, pp. 2364–2368.

[13] Y. Zheng and L. A. Tuan, "Incorporating template-based contrastive learning into cognitively inspired, low-resource relation extraction," *Cognitive Computation*, vol. 16, no. 6, pp. 3228–3240, 2024.

[14] G. Pleiss, T. Zhang, E. Elenberg, and K. Q. Weinberger, "Identifying mislabeled data using the area under the margin ranking," *Advances in Neural Inf. Processing Systems*, vol. 33, pp. 17 044–17 056, 2020.

[15] T. Sosea and C. Caragea, "Leveraging training dynamics and self-training for text classification," in *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022, pp. 4750–4762.

[16] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[17] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer, "Rethinking the role of demonstrations: What makes in-context learning work?" *arXiv preprint arXiv:2202.12837*, 2022.

[18] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proc. of the 2019 conf. of the North American chapter of the association for computational linguistics: human language technologies, vol. 1*, 2019, pp. 4171–4186.

[20] D. Zhang and D. Wang, "Relation classification via recurrent neural network," *arXiv preprint arXiv:1508.01006*, 2015.

[21] L. B. Soares, N. FitzGerald, J. Ling, and T. Kwiatkowski, "Matching the blanks: Distributional similarity for relation learning," *arXiv preprint arXiv:1906.03158*, 2019.

[22] Z. Wan, F. Cheng, Z. Mao, Q. Liu, H. Song, J. Li, and S. Kurohashi, "Gpt-re: In-context learning for relation extraction using large language models," *arXiv preprint arXiv:2305.02105*, 2023.

[23] T. Gao, X. Yao, and D. Chen, "Simcse: Simple contrastive learning of sentence embeddings," *arXiv preprint arXiv:2104.08821*, 2021.

[24] I. Hendrickx, S. N. Kim, Z. Kozareva, P. Nakov, D. O. Séaghdha, S. Padó, M. Pennacchiotti, L. Romano, and S. Szpakowicz, "Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals," *arXiv preprint arXiv:1911.10422*, 2019.

[25] Y. Zhang, V. Zhong, D. Chen, G. Angeli, and C. D. Manning, "Position-aware attention and supervised data improve slot filling," in *Conference on empirical methods in natural language processing*, 2017.

[26] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-supervised self-training of object detection models," 2005.

[27] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in NeurIPS*, vol. 30, 2017.

[28] H. Lin, J. Yan, M. Qu, and X. Ren, "Learning dual retrieval module for semi-supervised relation extraction," in *The world wide web conference*, 2019, pp. 1073–1083.

[29] W. Li and T. Qian, "Exploit multiple reference graphs for semi-supervised relation extraction," *arXiv preprint arXiv:2010.11383*, 2020.

[30] X. Hu, C. Zhang, Y. Yang, X. Li, L. Lin, L. Wen, and P. S. Yu, "Gradient imitation reinforcement learning for low resource relation extraction," *arXiv preprint arXiv:2109.06415*, 2021.

[31] B. Mao, C. Jia, Y. Huang, K. He, J. Wu, T. Gong, and C. Li, "Uncertainty-guided mutual consistency training for semi-supervised biomedical relation extraction," in *2022 IEEE Int. Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2022, pp. 2318–2325.

[32] S. Zhang, D. Zheng, X. Hu, and M. Yang, "Bidirectional long short-term memory networks for relation classification," in *Proc. of the 29th Pacific Asia conf. on language, inf. and computation*, 2015, pp. 73–78.

[33] D. Zeng, K. Liu, Y. Chen, and J. Zhao, "Distant supervision for relation extraction via piecewise convolutional neural networks," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 1753–1762.

[34] G. Nan, Z. Guo, I. Sekulić, and W. Lu, "Reasoning with latent structure refinement for document-level relation extraction," *arXiv preprint arXiv:2005.06312*, 2020.

[35] X. Hu, C. Zhang, Y. Xu, L. Wen, and P. S. Yu, "Selfore: Self-supervised relational feature learning for open relation extraction," *arXiv preprint arXiv:2004.02438*, 2020.

[36] W. Li and T. Qian, "Exploit multiple reference graphs for semi-supervised relation extraction," *arXiv preprint arXiv:2010.11383*, 2020.

[37] X. Li, Q. Sun, Y. Liu, Q. Zhou, S. Zheng, T.-S. Chua, and B. Schiele, "Learning to self-train for semi-supervised few-shot classification," *Advances in neural information processing systems*, vol. 32, 2019.

[38] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *NeurIPS 2020*, vol. 33, pp. 1877–1901, 2020.

[39] A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, C. Li, C. Li, D. Liu, F. Huang *et al.*, "Qwen2 technical report," *arXiv preprint arXiv:2407.10671*, 2024.

[40] M. Agrawal, S. Hegselmann, H. Lang, Y. Kim, and D. Sontag, "Large language models are few-shot clinical information extractors," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022, pp. 1998–2022.

[41] X. Wei, X. Cui, N. Cheng, X. Wang, X. Zhang, S. Huang, P. Xie, J. Xu, Y. Chen, M. Zhang *et al.*, "Zero-shot information extraction via chatting with chatgpt," *arXiv preprint arXiv:2302.10205*, 2023.

[42] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *NeurIPS 2022*, vol. 35, pp. 24 824–24 837, 2022.

[43] J. Liu, D. Shen, Y. Zhang, B. Dolan, L. Carin, and W. Chen, "What makes good in-context examples for gpt-3?" *arXiv preprint arXiv:2101.06804*, 2021.

[44] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," *arXiv preprint arXiv:2203.11171*, 2022.

[45] J. Liu, D. Shen, Y. Zhang, B. Dolan, L. Carin, and W. Chen, "What makes good in-context examples for gpt-3?" *arXiv preprint arXiv:2101.06804*, 2021.

[46] R. Shin, C. H. Lin, S. Thomson, C. Chen, S. Roy, E. A. Platanios, A. Pauls, D. Klein, J. Eisner, and B. Van Durme, "Constrained language models yield few-shot semantic parsers," *arXiv preprint arXiv:2104.08768*, 2021.

[47] O. Rubin, J. Herzig, and J. Berant, "Learning to retrieve prompts for in-context learning," *arXiv preprint arXiv:2112.08633*, 2021.

[48] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.