2024

Few-shot Learning for NER using MAML

Nourchene Bargaoui
Virginia Commonwealth University

Few-shot Learning for NER using MAML

NOURCHENE BARGAOUI

A thesis submitted in partial fulfilment of the requirements

for the degree of

Master of Science at Virginia Commonwealth University

Principal Supervisor:

Prof. Bridget T. McInnes (Virginia Commonwealth University)

May 2024

DECLARATION

I hereby declare that this thesis represents my own work which has been done after registration for the degree of Master's at Virginia Commonwealth University, and has not been previously included in a thesis or dissertation submitted to this or any other institution for a degree, diploma or other qualifications.

I have read the University's current research ethics guidelines, and accept responsibility for the conduct of the procedures in accordance with the University's Research Ethics Committee (REC). I have attempted to identify all the risks related to this research that may arise in conducting this research, obtained the relevant ethical and/or safety approval (where applicable), and acknowledged my obligations and the rights of the participants.

Signature:___________________

May 2024

Abstract

This thesis investigates the application of Few-Shot Learning (FSL) using Model-Agnostic Meta-Learning (MAML) to enhance Named Entity Recognition (NER) within the domain of Natural Language Processing (NLP), specifically focusing on chemical datasets. The primary challenge addressed is the impracticality of relying on extensive annotated datasets, especially in specialized fields like chemistry. The research primarily explores the concept of Few-Shot Learning, aiming to train models on minimal data while maintaining performance across diverse tasks. It delves into the N-way K-shot methodology, where "N" represents the number of classes and "K" signifies the number of examples per class. This approach is further investigated through the MAML method, a meta-learning strategy enabling models to quickly adapt to new tasks using only a few training examples. Key contributions of the thesis include the development of a comprehensive methodological framework employing MAML for NER tasks within the chemical context, demonstrated through experiments conducted on the ChEMU dataset. The challenges associated with applying FSL in NER are systematically presented, and an innovative solution is proposed through the adoption of the MAML method. The findings suggest that while FSL may not consistently outperform traditional models with large datasets, it offers a compelling alternative in scenarios where data is limited. This has significant implications for future research in NLP applications, particularly in specialized domains like chemistry.

Keywords: BERT, NLP, Few shot Learning, MAML, BiLSTM

Acknowledgments

I would like to express my deepest gratitude to Dr. Bridget McInnes, my advisor, for her unwavering support and guidance throughout my journey at VCU. Her expertise and insights have been invaluable, not only to this thesis but to my overall growth as a scholar.

I extend my appreciation to my thesis committee, Dr. Tomaz Arodz, Dr. Tamer Nadeem and Dr. Christina Tang, for agreeing to evaluate my work on this thesis.

I am also profoundly thankful to the Fulbright Program for providing the funding necessary to pursue my Master's degree. This opportunity has been pivotal in advancing my academic and professional aspirations.

Lastly, I wish to acknowledge the support and encouragement from my peers and family, whose constant encouragement and faith in my abilities have been a source of motivation and strength.

Thank you all for your support, without which this journey would not have been possible.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

Named Entity Recognition (NER) is a fundamental task in Natural Language Processing (NLP) that involves identifying and classifying named entities within text. Named entities are specific pieces of information that are referred to by proper names, such as the names of people, organizations, locations, dates, quantities, and monetary values. The primary goal of NER is to accurately identify and classify these named entities into predefined categories. This task is crucial for various NLP applications, including information retrieval, document summarization, question answering, and sentiment analysis. However, NER poses several challenges due to the ambiguity and variability of natural language, as well as the diverse types and contexts of named entities. As a result, developing effective NER models requires sophisticated algorithms and techniques that can accurately identify named entities across different languages, domains, and textual genres.

Transitioning from the broader challenges of NER, this work focuses on addressing the specific difficulties encountered in analyzing chemical datasets. Chemical data, characterized by its complexity and specificity, often suffers from a scarcity of labeled examples, posing significant challenges for conventional machine learning techniques.

In response to these challenges, we propose a modern approach that integrates Few-Shot Learning (FSL) with the Model-Agnostic Meta-Learning (MAML) algorithm. By leveraging FSL and MAML, our model demonstrates the capability to rapidly adapt to new tasks with minimal data, thereby reducing dependency on extensive annotated datasets. This approach not only enhances the generalization ability of the model across different chemical tasks but also opens avenues for more efficient utilization of available data.

Furthermore, our work contributes to the broader landscape of machine learning in chemistry by providing insights into the potential applications of FSL and MAML in addressing data scarcity challenges. Through rigorous evaluation and experimentation, we aim to not only advance the field of NER in specialized domains such as chemistry but also facilitate the broader application of machine learning techniques in addressing real-world problems.

The thesis is organized as follows. First, we discuss the background and related work. Second, we present our proposed method including the dataset and evaluation methodology. Third, we discuss our experimental results and conclusions. Fourth, we discuss our future work. Lastly, we describe our contributions.

Chapter 2

Related Work

The few-shot learning problem, integral to adapting machine learning models to new tasks or domains with minimal labeled examples, is commonly conceptualized as an N-way K-shot problem. Each task comprises N classes, with K examples of each class forming the support set used for model training, while performance is evaluated on a separate query set. In the realm of natural language processing (NLP), various methodologies have been employed to tackle the few-shot learning challenge, with meta-learning and data augmentation emerging as the main strategies. Meta-learning approaches for NLP encompass metric-based, optimization-based, and hybrid methods, as delineated in a comprehensive survey paper by Parnami et al. [16] Metric-based and optimization-based techniques have been recognized and explored since 2018, whereas the hybrid approach, which integrates elements of both, has recently gained attention through literature review. To provide clarity, Figure 2.1 illustrates the timeline of these methodologies, categorizing metric-based, optimization-based, and hybrid approaches separately from meta-learning, despite their conceptual overlap. Conversely, data augmentation for few-shot learning is a comparatively nascent area, making its debut in 2022.

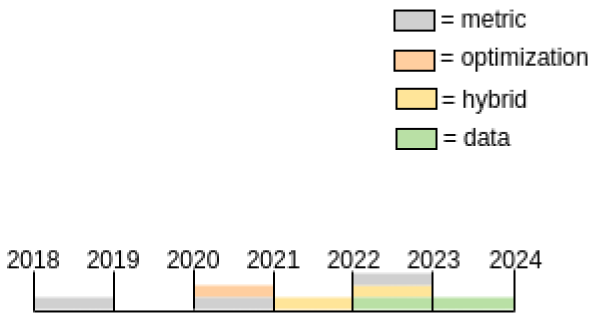

Figure 2.1: Timeline of different few-shot learning methodologies in NLP

Meta-learning, a subtype of few-shot learning, revolves around the concept of 'learning to learn.' In this paradigm, a meta-learner is tasked with discovering or learning the optimal learning algorithms, known as learners, for various tasks. Each task within the meta-learning framework comprises a support set containing k examples, used for training the learner, and a query set employed to evaluate the performance of the model on unseen data. On the other hand, data augmentation involves the manipulation or generation of data to expand the existing dataset, thereby enhancing the model's ability to generalize across diverse instances. While data augmentation has been a well-established practice in machine learning, its application to few-shot learning in NLP is relatively new, with only a handful of papers published in 2022 and 2023.

In the following subsections, we delve into each of these few-shot learning methods in more detail, examining their underlying principles, methodologies, and applications in the context of Natural Language Processing.

2.1 Meta-learning

In general, metric-based approaches in few-shot learning aim to leverage the distance between vector representations of input data for classification. This category encompasses various methodologies, including Matching Networks [18], Siamese Networks [10], and Prototypical Networks [17], which exploit similarities or dissimilarities between samples to make predictions.

On the other hand, optimization-based approaches focus on learning the optimal parameters for adapting to new tasks quickly. Notable examples in the realm of NLP include Model Agnostic Meta-Learning (MAML) [6], LEOPARD [1], and REPTILE [14], with MAML being a prominent representative of this approach. These methods aim to anticipate and adjust model parameters in a way that facilitates rapid adaptation to new tasks.

Hybrid approaches, as the name suggests, integrate elements of both metric-based and optimization-based techniques. While fewer in number compared to the other two methods, hybrid approaches represent a promising avenue for meta-learning in NLP. Although there are no seminal examples, this thesis will delve into some of these hybrid approaches in greater detail in subsequent sections, exploring their potential and effectiveness in few-shot learning scenarios.

Finally, data augmentation methods play a crucial role in alleviating the scarcity of labeled data by manipulating existing data or generating synthetic data samples. While the concept of data augmentation is well-established in machine learning, its application to few-shot learning in NLP is still in its infancy, with limited research and no widely recognized benchmarks or methodologies.

2.1.1 Metric-based

In the realm of few-shot learning, metric-based approaches offer a diverse array of methodologies aimed at leveraging the similarities or dissimilarities between samples for classification. We identified several papers that utilize different metric-based methods, including Matching Networks [18], Siamese Neural Networks [10], Structured Nearest Neighbor techniques, and Prototypical Networks [17]. While each method varies in its implementation, they all share the fundamental principle of relying on effective representation learning for accurate classification.

Matching Networks [18] employ two embedding functions and an attention mechanism to determine the 'closest' class for a given query instance. This attention mechanism computes the softmax of the cosine similarity between the query instance embedding and the class embedding representations generated from the support set examples. Yu et al. [20] further enhance Matching Networks by introducing a task clustering method, allowing for different Matching Network setups within each cluster.

Siamese Neural Networks, introduced by Koch et al. (2015), utilize twin networks that compute the similarity between outer-level feature representations of different inputs. Oniani et al. [15] adapt this approach for clinical NLP tasks, demonstrating its effectiveness in few-shot learning scenarios.

Yang, et al. [19] propose a structured nearest neighbor approach, where the class of the closest instance in vector space from the support set is assigned to the query instance. This structured approach incorporates additional information from a Conditional Random Field (CRF), ensuring generalizability across various tasks.

Prototypical Networks [17] generate prototypes or average embedding representations for each class, which are then compared against the embedding representation of each instance in the query set. Various distance metrics can be utilized for this comparison. Fritzler et al. (2018) apply Prototypical Networks to Named Entity Recognition tasks, employing the Euclidean squared distance for comparison.

2.1.2 Optimization-based

Optimization-based approaches to few-shot learning predominantly rely on the Model-Agnostic Meta-Learning (MAML) framework [6], which serves as the cornerstone of this methodological category. MAML operates by leveraging the negative gradient of a loss function, where the gradient represents the direction of steepest increase and its negation signifies the direction of steepest decrease. Initially, MAML initializes model parameters randomly and then performs gradient steps, adjusting these parameters based on the support set of each training task. Subsequently, it computes the gradients of the losses of these adapted parameters on the query set and aggregates them to update the original parameters.

One notable extension of MAML is LEOPARD [11], which introduces task-specific layers alongside task-agnostic layers. LEOPARD's distinguishing feature is its inclusion of a task-specific softmax layer, allowing for varying numbers of classes per task—a departure from MAML's uniform class structure. Notably, LEOPARD has been specifically tailored for NLP tasks, making it an ideal choice for few-shot learning in this domain.

Several studies have adopted LEOPARD for NLP tasks, leveraging its capabilities for effective few-shot learning. For instance, Bansal et al. [2] utilize LEOPARD in a self-supervised setting, where tasks are generated from the English Wikipedia dump. Additionally, they partition the original Named Entity Recognition (NER) dataset into multiple tasks, further showcasing the versatility and applicability of LEOPARD in various NLP scenarios.

2.1.3 Hybrid

Hybrid approaches to few-shot learning represent a fusion of metric-based and optimization-based techniques, leveraging the strengths of both methodologies to enhance model performance. In our review, we encountered two papers that exemplify this hybrid approach, both of which utilize Prototypical Networks [17] in combination with variations of the Model-Agnostic Meta-Learning (MAML) framework [6].

The first paper [12] adopts a hybrid strategy for Named Entity Recognition (NER) by decomposing the task into two subtasks: span detection and entity typing. For span detection, they employ First-Order MAML (FOMAML), a computationally efficient approximation of MAML that significantly reduces the computational overhead associated with second-order derivatives. Meanwhile, for entity typing, which involves labeling entity spans within text, they employ a combination of Prototypical Networks and FOMAML. Specifically, they use regular gradient descent in an inner loop to obtain copies of the embedding function parameters for each task, which are then aggregated in a FOMAML manner to update the original embedding function parameters.

The second paper[4] adopts a similar hybrid approach but with the REPTILE algorithm (Nichol et al., 2018), another first-order approximation for MAML. One notable advantage of REPTILE is its ability to dispense with the need for dividing data into support and query sets during the meta-learning phase. Leveraging this characteristic, the authors introduce the ProtoREPTILE algorithm, which utilizes the support set solely for creating representations and employs the query set alone for the optimization-based approach. This streamlined approach demonstrates the versatility and effectiveness of hybrid methodologies in few-shot learning scenarios.

2.2 Data Augmentation

Data augmentation, a novel approach in the realm of few-shot learning in NLP, involves the manipulation or generation of data to enrich existing datasets. Despite its recent emergence, this category has garnered attention, with only a limited number of papers published in 2022 and 2023.

Chen et al. [3] proposed a data augmentation method that involves replacing entities in sentences with adjacent terms sourced from biomedical Unified Medical Language System (UMLS) knowledge graphs. They subsequently treat the Named Entity Recognition (NER) problem as a Question Answering (QA) task, leveraging a pretrained BioBERT model. This approach preserves the labels of the original data while augmenting the dataset with semantically similar instances.

In contrast, FlipDA [21] explored a different approach to data augmentation by observing that flipping the labels of sentences can improve performance on SuperGLUE tasks using large pretrained models. Unlike the approach by Chen et al. [3], FlipDA alters the labels of sentences during data augmentation. They achieve this by incorporating techniques such as word addition, deletion, or substitution, along with label flipping. For data generation, FlipDA utilizes T5, a text-to-text transformer model, to mask certain words, which are then replaced with T5's output when the label is flipped.

While both approaches aim to augment limited data to provide more examples for model training, they differ in their treatment of labels — Chen et al. [3] preserve labels, whereas FlipDA ([21] changes labels. These contrasting methodologies highlight the versatility of data augmentation techniques in addressing the challenges of few-shot learning in NLP.

2.3 Tasks

Table 2.1 presents a comparison of different approaches applied to various Natural Language Processing (NLP) tasks. The "NLP Tasks" column lists specific NLP tasks such as Sentiment Classification, Intent Classification, Named Entity Recognition (NER), Entity Typing, Task Classification, Natural Language Inference (NLI), Question Answering (QA), Textual Entailment, Coreference Resolution, Word-Sense Disambiguation (WSD), and Text Classification. The "Metric", "Optimization", "Hybrid", and "Data" columns represent different approaches applied to each NLP task. Entries in these columns correspond to references (citations) of papers or studies that utilized the respective approach for the corresponding task.

Overall, the table shows that multiple approaches are applied to different NLP tasks, with variations depending on the nature of the task and the requirements of the application. Some tasks have multiple approaches applied to them, while others have only one or none. There is a slight overlap between metric and optimization-based approaches in tasks such as Sentiment Classification, indicating that different methodologies have been explored for addressing similar tasks. Data augmentation approaches, represented in the "Data" column, have been a subset of tasks including QA, Textual Entailment, Coreference Resolution, and WSD, as shown by the presence of citations in these rows.

Table 2.1: Comparison of Applications

NLP Tasks	Metric	Optimization	Hybrid	Data
Sentiment	[20]	[1][2]		
Intent	[20]			
NER	[15][19][7]	[11]	[4][12]	[3]
Entity Typing		[2][1][12]		
Task Classification		[1][2]		
NLI		[1][2]		
QA				[21]
Textual Entailment				[21]
Coref. Resolution				[21]
WSD				[21]
Text Classification	[20]			

2.4 Discussion

The exploration of few-shot learning in the NLP domain reveals two main categories: meta-learning and data augmentation. Within meta-learning, three primary approaches emerge: metric-based, optimization-based, and hybrid. Metric-based approaches encompass Prototypical Networks, Siamese Networks, Matching Networks, and Nearest Neighbor techniques. On the other hand, optimization-based approaches, particularly focused on NER and NLP tasks, show a preference for LEOPARD, a method based on the Model-Agnostic Meta-Learning (MAML) framework. Notably, hybrid approaches combine MAML approximations with Prototypical Networks, indicating a convergence of methodologies in addressing few-shot learning challenges.

However, the current literature predominantly emphasizes general domain applications over specialized domains, despite the potential benefits few-shot learning could offer in specialized contexts. Moreover, while there is considerable research attention on general NLP tasks such as Sentiment Classification and Question Answering, there is a notable lack of systems specifically tailored for Named Entity Recognition (NER).

Further exploration is warranted in several areas. Firstly, while hybrid methods predominantly focus on Prototypical Networks, investigating the applicability of other metric-based approaches could yield valuable insights. Secondly, empirical evidence suggests that increasing the number of shots or examples improves performance in meta-learning tasks, prompting consideration for integrating data augmentation techniques to enhance meta-learning effectiveness. Although limited literature exists on the combination of meta-learning with data augmentation, promising avenues are evident, as demonstrated by a recent paper [13] that utilized both techniques. This interdisciplinary approach presents a promising direction for future research in optimizing few-shot learning performance.

Chapter 3

Methodology

3.1 Research Goals and Experimental Design

In our review of the literature, we've identified a significant gap in research concerning the application of few-shot learning techniques for Named Entity Recognition (NER) within the chemical domain. Existing studies primarily focus on well-defined English entities, overlooking the challenges posed by limited labeled data and the need for rapid adaptation to new entity types and contexts specific to chemistry. To address this gap, we aim to leverage Model-Agnostic Meta-Learning (MAML) to enhance NER performance in scenarios with sparse annotations and diverse entity distributions within the chemical domain.

Research Question #1: How do varying shot sizes impact the performance and adaptability of MAML-based NER models in handling limited data scenarios? To answer this question, we propose a comprehensive experimental study aimed at gaining insights into the behavior of MAML-based NER models across various experimental settings. Our study systematically investigates the impact of varying shot sizes and task distributions on the performance of MAML-based NER models. By exploring the behavior of these models under diverse experimental conditions, we aim to assess their scalability and adaptability to different data regimes. Specifically, we will vary the shot sizes, ranging from very few examples per class to larger support sets, to understand how the model's performance is affected by the availability of training data. Additionally, we will analyze different task distributions to evaluate the model's flexibility across varied contexts, such as the CHEMU dataset, which encompasses distinct entity types within the chemical domain. Through this focused analysis, we seek to gain valuable insights into the behavior of MAML-based NER models and their suitability for handling limited data scenarios in real-world applications.

Research Question #2: How transferable are the learned representations and adaptation strategies acquired by MAML-based NER models across closely related and distinct entity types within the chemical domain? To answer this question, we propose to evaluate the transferability of MAML-based NER models by testing their performance on closely related entity types within the chemical domain, such as compound names and reaction steps, as well as distinct entity types, such as reaction products. Specifically, we will assess the effectiveness of the learned representations and adaptation strategies acquired through MAML training when applied to the extraction of these chemically related entities, which may exhibit overlapping or interconnected characteristics. This evaluation will provide valuable insights into the transferability of the learned representations and adaptation strategies, shedding light on the generalization capabilities of MAML-based NER models across closely related and distinct entity types within the chemical domain.

3.2 Named Entity Recognition System

Our base NER system leverages Bidirectional Encoder Representations from Transformers (BERT) to generate contextualized embedding representations, which are then passed into Bidirectional Long Short-Term Memory units (biLSTMs).

BERT is a pretrained model trained on a large corpus for tasks such as masked language modeling and next sentence prediction. Devlin et al. [5] demonstrated that fine-tuning BERT for specific NLP tasks, including NER, by adding a classification layer yields effective performance.

Long Short-Term Memory units (LSTMs), introduced by Huang et al.[9], are a form of recurrent neural networks known for their ability to capture long-range dependencies in sequential data. LSTMs incorporate both current and past states as input, allowing them to establish connections between past observations, such as words in a sentence, and learn dependencies over extended distances. Their gating mechanism enables them to regulate the flow of information, deciding which information to retain and pass on to subsequent components, thus facilitating the extraction of relevant features.

In our architecture, bidirectional LSTMs process data in both forward and backward directions through two separate hidden layers, which are then merged and fed into a common output layer. This bidirectional processing enables the model to capture contextual information from both preceding and succeeding tokens in the input sequence, enhancing its understanding of the overall context. Finally, we apply a softmax layer to the output of the biLSTM to compute the final class probability for each token in the sequence.

For a visual representation of our system architecture, refer to Figure 3.1, which provides a high-level overview of the complete system.

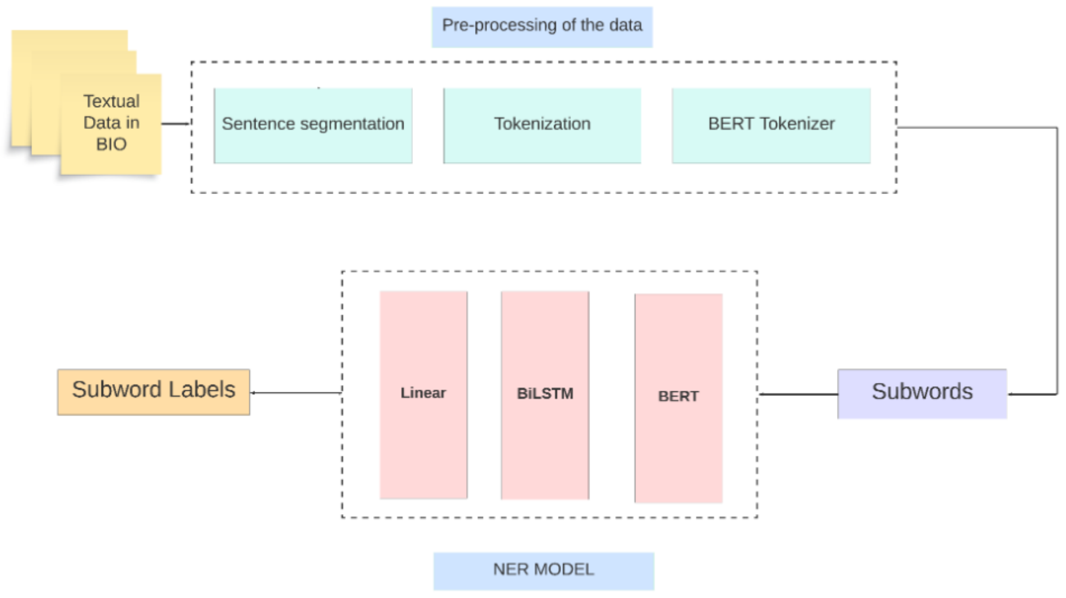

Figure 3.1: Basic Named Entity Recognition System Diagram

This NERModel is a neural network for NER specifically tailored for the CHEMU dataset, utilizing both a pre trained BERT model and a BiLSTM layer followed by a linear layer. Here's a breakdown of the model components and their roles:

BERT Model Initialization: The model uses BertModel from pretrained(bert base uncased) to load a pre trained BERT model with the base-uncased configuration. This part of the model serves as the feature extractor, converting input text into a sequence of contextual embeddings. Each token of the input text is represented by a 768-dimensional vector.

Dropout Layer: A dropout layer with a dropout rate of 0.1 is applied to the output of the BERT model. This is a regularization technique used to prevent overfitting by randomly setting a fraction of the input units to 0 at each update during training.

BiLSTM Layer: Following the dropout layer, the model employs a bidirectional LSTM (BiLSTM) with two layers (number of layers = 2). Each direction of the LSTM has a hidden size of 100, making the total output for each time step 200 (100 for each direction). The BiLSTM processes the sequence output from BERT, capturing dependencies in both forward and backward directions along the sequence. This is particularly useful in NER to utilize both past and future context.

Linear Layer: The final component is a linear layer that maps the 200 dimensional output from the BiLSTM to the number of unique labels (num labels) in the NER task. This layer generates logits for each label, which can be interpreted as raw, unnormalized scores for each label class.

Forward Method: The forward method defines how the data flows through the model. It takes input ids and attention mask as inputs, which are standard inputs for BERT models to handle tokenized text and ignore padding respectively. The method proceeds by passing inputs through the BERT model, applying dropout to the BERT output, processing it through the BiLSTM, and finally generating logits with the linear layer. These logits are typically passed through a softmax function during training or inference to obtain probability distributions over the label classes.

3.3 Few shot learning Framework

In this work, we have developed a Model-Agnostic Meta-Learning (MAML) framework to explore few-shot learning for Named Entity Recognition (NER). MAML, introduced by Finn et al., is a meta-learning algorithm designed to enable models to quickly adapt to new tasks with minimal data, offering a promising solution to the data-intensive nature of NER models.

The MAML framework consists of three distinct phases: meta-training, meta-testing, and meta-development, which are not depicted in the diagram. Each phase encompasses unique tasks with non-overlapping support sets containing k shots or examples and a query set used for adjustments to the meta-learner. In the meta-training phase, we begin with a randomly initialized set of parameters for a generalized model, referred to as the meta-learner. Copies of the meta-learner, known as learners, are created and trained on the support set. Subsequently, the learners are evaluated on the query set using a loss function—in our case, Binary Cross-Entropy Loss with the Adam optimizer. The losses across tasks are then averaged and backpropagated through the meta-learner to update its parameters.

Similarly, in the meta-development and meta-testing phases, the same process is applied, with the exception that there are no updates to the meta-learner. Instead, only the learners are updated based on the support set and evaluated on the query set for inference.

This method represents a promising strategy to overcome the limitations posed by datasets with few labeled examples, a common scenario in specialized domains such as chemistry. By leveraging the Few-Shot Learning paradigm through MAML, our model aims to generalize effectively to new tasks, thereby reducing the dependency on large annotated datasets and enabling more efficient utilization of available data.

To integrate our Named Entity Recognition (NER) system described in Section 3.2, we decompose the NER task into N 2-way classification problems. For example, if the entities of interest are 'STARTING MATERIAL,' 'REACTION STEP,' and 'REACTION PRODUCT,' we break them down into three binary classification problems: 'STARTING MATERIAL' versus 'not a STARTING MATERIAL,' 'REACTION STEP' versus 'not a REACTION STEP,' and 'REACTION PRODUCT' versus 'not a REACTION PRODUCT.' This decomposition is necessary because MAML requires multiple tasks to generalize effectively.

The authors, Finn et al. in 2017 [6], introduced a meta-learning algorithm that is adaptable to any model trained via gradient descent, distinguishing it from other methods. The effectiveness of their approach is demonstrated through various experiments across multiple domains. These experiments show that training with a minimal number of data points and a limited number of gradient steps can indeed yield significant generalization on specific tasks, denoted as T (as shown in the algorithm below). The algorithm is designed to identify optimal initial model parameters that facilitate robust outcomes across a diverse set of tasks T. This is achieved by fine-tuning the initial parameters to enhance adaptability to new tasks. This innovative approach is illustrated in Figure 3.2. The methodology was considered revolutionary, providing unprecedented flexibility in meta-learning. It opened up possibilities for generalizing across a broad spectrum of tasks, ranging from simple linear models to complex challenges like image classification.

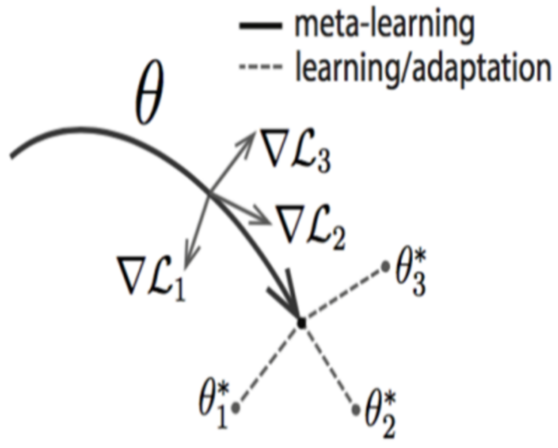

Figure 3.2: Image from Joshua Ball , Few-Shot Learning for Image Classification of Common Flora, May 2021. Here we see how MAML chooses its next θ for its next learning and adaption step in gradient descent.

As we explained in Chapter 1, meta-learning approaches are categorized into three types: metric-based, optimization-based, and model-based. In this section, we will focus on the mathematical principles behind optimization-based meta-learning methods.

Terminology: Meta-learning models undergo training using a meta-training dataset comprising a series of tasks, denoted as $\mathcal{T} = \{\tau_1, \tau_2, \tau_3, \ldots\}$. These models are then evaluated using a meta-testing dataset consisting of tasks τ_{ts}. Each task τ_i includes a task training set, also known as the support set D_i^{tr}, and a task test set, or query set D_i^{ts}. A common scenario in meta-learning is the N-way k-shot learning, where the objective is to differentiate between N classes, learning with k examples from each class.

Transfer learning (fine-tuning): Before delving into meta-learning, it is important to touch on another prevalent strategy which is transfer learning through fine-tuning. This method involves transferring knowledge from a base model, typically trained to recognize different objects, to a more specialized task, such as identifying specific animals like dogs or cats. The core concept here involves using models that are pre-trained on general tasks and then fine-tuning them for specific applications. This fine-tuning process usually entails updating a limited number of layers within the neural network and/or adjusting to a slower learning rate to refine the model's performance on the new task.

In a typical fine-tuning scenario, the process begins with a set of pre-trained parameters, θ_{pre-tr}, optimized on a pre-training dataset, D^{pre-tr}.

$$\theta_{\text{pre-tr}} = \theta_0 - \alpha \nabla \mathcal{L}(\theta, D^{\text{pre-tr}})$$

During fine-tuning, we would then tune the parameters that minimize the loss to training set D^{tr}.

$$\theta = \theta_{\text{pre-tr}} - \alpha \nabla \mathcal{L}(\theta, D^{\text{tr}})$$

The equation illustrates one gradient step, but in practice this is optimized via multiple gradient steps.

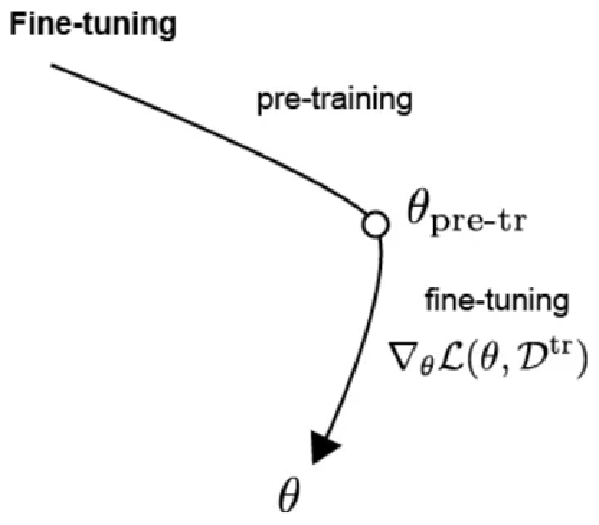

Figure 3.3: As an illustration, below shows the paths in the parameter space going from the pre-trained parameter values $\theta_{\text{pre-tr}}$ toward the fine-tuned parameter values θ.

In transfer learning via fine-tuning, the hope is that the base model have learned the basic patterns (such as shapes, contrasts, objects in images) that fine-tuning can more quickly and easily adopt to a new task. However, the approach is not specifically designed explicitly around learning to learn. The novel task may not overlap with the base tasks and result in poor performance for the transfer of knowledge. Meta-learning, on the other hand, is designed explicitly around constructing tasks and algorithms for generalizable learning.

MAML: This is an optimization-based meta-learning approach. The idea is that instead of finding parameters that are good for a given training dataset or on a fine-tuned training set, we want to find optimal parameters that with fine-tuning are generalizable to other test sets. In Algorithm 1, we present the pseudo-code for MAML. Initially, the model is randomly initialized with parameters θ. For each task sample, we calculate the loss, compute the gradient, and update the parameters by subtracting it from θ multiplied by a constant α, saving the result into a separate variable θ'. We repeat this process for each task, creating a new parameter set θ_i for each. Finally, we evaluate the model with the updated parameters, compute the gradient of the combined loss, add them up, multiply by a constant β, and subtract from the original θ. In essence, we duplicate the model for each task, update the parameters, and aggregate them in the end, typically by summing up the gradients of the loss function.

Algorithm 1 MAML Training Algorithm

```
Input: $p(\mathcal{T})$: distribution over tasks
Input: $\alpha, \beta$: step size hyperparameters
1: Randomly initialize $\theta$
2: while not done do
3:     Sample batch of tasks $T_i \sim p(\mathcal{T})$
4:     for all $T_i$ do
5:         Evaluate $\nabla_\theta \mathcal{L}_{T_i}(f_\theta)$ with respect to $K$ examples
6:         Compute adapted parameters with gradient descent: $\theta_i = \theta - \alpha \nabla_\theta \mathcal{L}_{T_i}(f_\theta)$
7:     end for
8:     Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i \sim p(\mathcal{T})} \mathcal{L}_{T_i}(f_{\theta_i})$
9: end while
```

For one task, given a task, we will first use a support training dataset D^{tr} in a fine-tuning step. The optimal parameter θ' for D^{tr} for MAML, revolutionized the field with its optimization-centric approach to meta-learning. Rather than simply optimizing parameters for a specific training dataset or through fine-tuning alone, MAML seeks to discover parameters that exhibit strong generalization capabilities across various test sets. At its core, MAML operates by iteratively fine-tuning parameters on a support training dataset, denoted as D^{tr}, to derive an optimal parameter set θ'. This finely tuned parameter set is meticulously crafted to adapt swiftly to new tasks, fostering enhanced generalization and performance.

$$\theta' = \theta - \alpha \nabla_\theta \mathcal{L}(\theta, D^{tr})$$

MAML goes further to evaluate the performance of the optimized parameter set θ on a separate query test dataset, denoted as D^{ts}, using the loss function $\mathcal{L}(\theta, D^{ts})$ which is unlike traditional fine-tuning approaches, where the process typically halts. The overarching objective is to refine the initial parameter θ such that it demonstrates robust performance on the query test set post fine-tuning.

Algorithm 2 MAML

```
Input: $p(\mathcal{T})$: distribution over tasks
Input: $\alpha, \beta$: step size hyperparameters
 1: $\theta_0 \leftarrow$ random initialization
 2: for $j \leftarrow 1$ to $J$ do
 3:     Sample a batch of $I$ tasks randomly from $p(\mathcal{T})$
 4:     for $i \leftarrow 1$ to $I$ do
 5:         $D^i_{\text{train}}, D^i_{\text{test}} \leftarrow$ dataset for task $\mathcal{T}_i$ from $D_{\text{meta-train}}$
 6:         $\mathcal{L}^i_{\text{train}} \leftarrow \mathcal{L}(f(D^i_{\text{train}}, \theta_{j-1}))$   ▷ Get loss of learner on training data
 7:         $\theta_i \leftarrow \theta_{j-1} - \alpha \nabla_{\theta_{j-1}} \mathcal{L}^i_{\text{train}}$   ▷ Adapt to learner's training loss
 8:         $\mathcal{L}^i_{\text{test}} \leftarrow \mathcal{L}(f(D^i_{\text{test}}, \theta_i))$   ▷ Get test loss of learner on adapted parameters
 9:     end for
10:     $\theta_j \leftarrow \theta_{j-1} - \beta \nabla_{\theta_j - 1} \sum_{i=1}^{I} \mathcal{L}^i_{\text{test}}$   ▷ Update parameters on total test loss of learners
11: end for
```

In essence, this entails updating θ through a meta-training step, optimizing it to effectively adapt and excel across various tasks, thereby ensuring superior performance in real-world scenarios.

$$\theta = \theta - \beta \nabla_\theta \mathcal{L}(\theta', D^{ts})$$

Here we need to calculate $\nabla_\theta \mathcal{L}(\theta, D^{ts})$, which is the derivative of the loss function with respect to θ. We can illustrate the paths in the parameter space as follows :

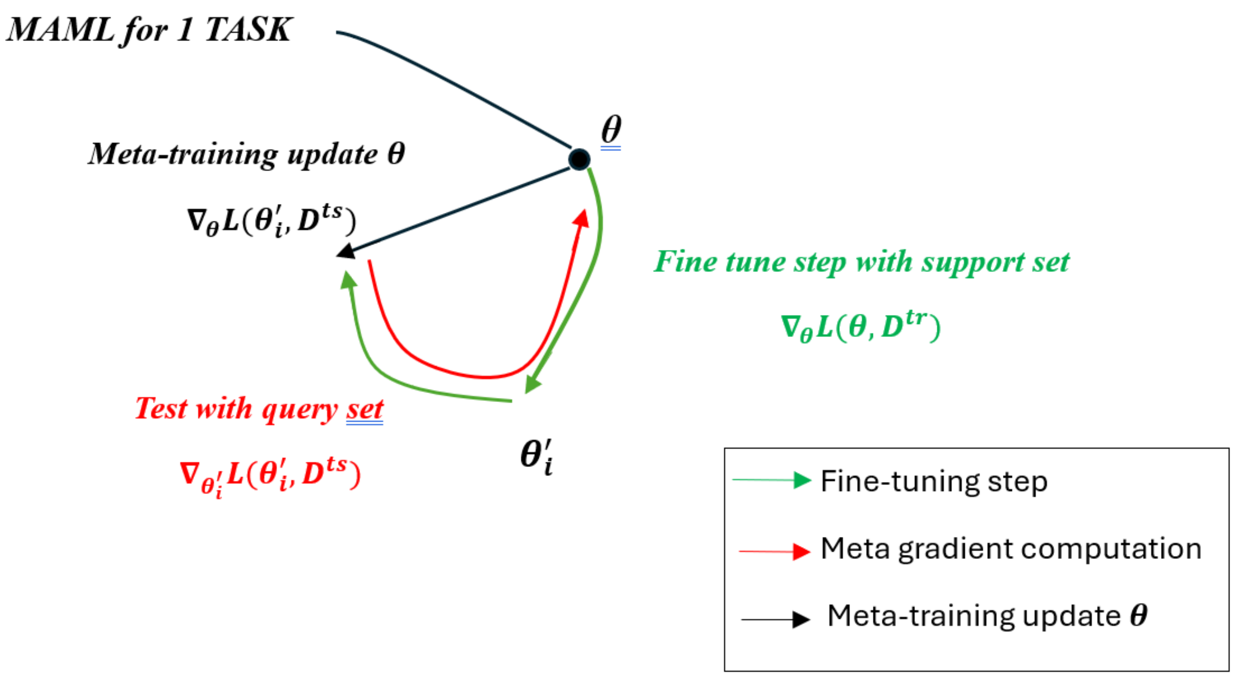
MAML for 1 TASK
θ
Meta-training update θ
$\nabla_\theta L(\theta_i', D^{ts})$
Fine tune step with support set
$\nabla_\theta L(\theta, D^{tr})$
Test with query set
$\nabla_{\theta_i'} L(\theta_i', D^{ts})$
θ_i'
Fine-tuning step
Meta gradient computation
Meta-training update θ

Figure 3.4: MAML Training Architecture

An important distinction lies in the methodology employed: rather than directly modifying θ during the fine-tuning phase, MAML gauges the trajectory towards optimal parameters by leveraging insights gleaned from both support train and test datasets (illustrated as paths in green), reserving the actual update of θ for the subsequent meta-training step. Expanding beyond individual tasks, MAML extends its reach to encompass task sets, aiming for broader generalizability across diverse task domains. This entails conducting meta-learning iteratively across a spectrum of tasks $\tau = \{\tau_1, \tau_2, \tau_3, \ldots\}$, thereby deriving optimal parameter sets θ_i' tailored to each task $\mathcal{T}i$ within the support set.

$$\theta_i' = \theta - \alpha \nabla \mathcal{L}(\theta, D_i^{\text{tr}})$$

The meta-training step is:

$$\theta = \theta - \beta \nabla_\theta \sum_i \mathcal{L}(\theta_i', D_i^{\text{ts}})$$

The term $\nabla_\theta \mathcal{L}(\theta_i', D_i^{\text{ts}})$ can be further expanded. Below we will omit the subscript i, but the discussion is applicable on a per-task basis. With the chain rule, the term can be expressed as:

$$\begin{aligned}\nabla_{\theta}\mathcal{L}(\theta', D^{\text{ts}}) &= \nabla_{\theta'}\mathcal{L}(\theta', D^{\text{ts}})\nabla_{\theta}\theta' \\ &= \nabla_{\theta'}\mathcal{L}(\theta', D^{\text{ts}})\nabla_{\theta}(\theta - \alpha\nabla_{\theta}\mathcal{L}(\theta, D^{\text{tr}})) \\ &= \nabla_{\theta'}\mathcal{L}(\theta', D^{\text{ts}})(I - \alpha\nabla_{\theta}^{2}\mathcal{L}(\theta, D^{\text{tr}}))\end{aligned}$$

We can expand on the earlier path visuals in the Figure 3.4, to include multiple tasks since for this thesis the MAML algorithm is multiple tasks. As illustrated in the Figure 3.5 below, the learning model aims to generalize well across tasks during meta-training, enabling it to quickly adapt to new tasks during meta-testing and meta-development without extensive retraining. The underlying idea behind MAML is its ability to leverage task-specific information from the support set to guide the parameter updates in a way that facilitates rapid adaptation to new tasks. By learning a set of meta-parameters that enable efficient task adaptation, MAML enables models to generalize well to new tasks with limited labeled data, making it particularly suitable for few-shot learning scenarios like NER.

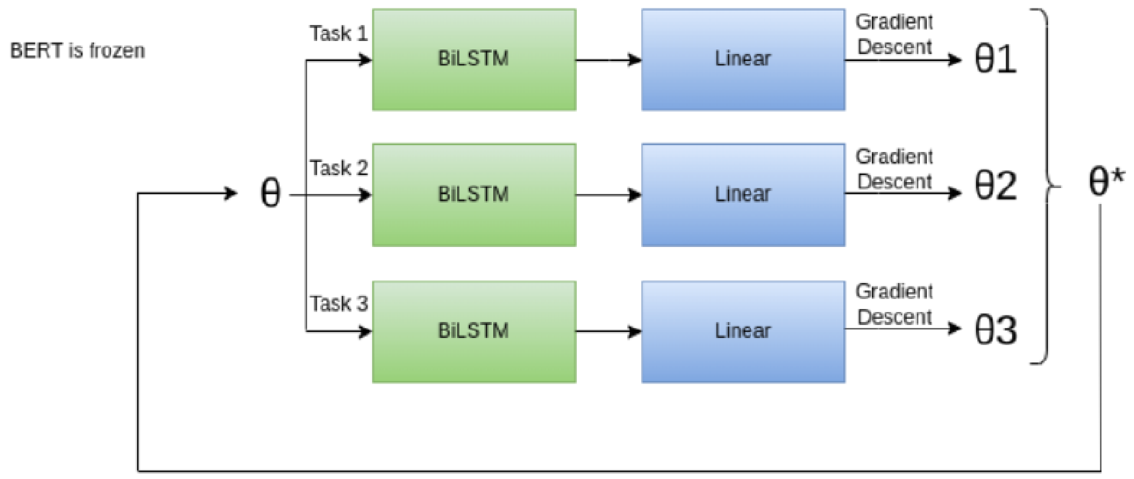

Figure 3.5: MAML Training Architecture

3.4 Dataset

The Cheminformatics Elsevier Melbourne University (ChEMU) data set [?] comprises 1500 chemical snippets extracted from 170 English patent documents sourced from both the European Patent Office and the United States Patent and Trademark Office. Each snippet provides a detailed description of chemical reactions. Entities in this dataset are categorized into four groups [8]: (1) chemical compounds involved in a chemical reaction; (2) conditions under which a chemical reaction occurs; (3) yields obtained for the final chemical product; and (4) example labels associated with reaction specifications. These four categories are further subdivided into a total of ten entity types. The compound category delineates five roles that a chemical compound can play within a chemical reaction. Both the conditions and yield categories include two entity types each.

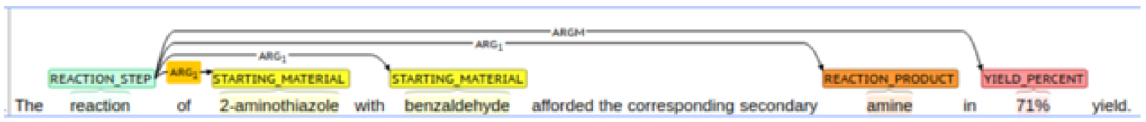
ARGM
ARG1
ARG1
REACTION_STEP
ARG1
STARTING_MATERIAL
STARTING_MATERIAL
REACTION_PRODUCT
YIELD_PERCENT
The reaction of 2-aminothiazole with benzaldehyde afforded the corresponding secondary amine in 71% yield.

Figure 3.6: Example of the CHEMU dataset

A chemical reaction step involves an action and one or more chemical compounds upon which the action takes effect. This action is also associated with the conditions under which it occurs and the resulting yields. Relations are established between actions (trigger words) and all the associated arguments in the reaction steps, including chemical compounds, conditions, and yields. The ARG1 event label corresponds to relations between a trigger word and chemical compound entities, while the ARGM event label corresponds to relations between a trigger word and temperature, time, or yield entities. Definitions of the entity types, trigger words, and relation types are provided in Table 3.1. Figure 3.6 presents an example instance.

Table 3.2 shows the number of entities in the Train, Development, and Test split.

Table 3.1: Definitions of entity types, trigger words, and relation types of CLEF-2020 dataset [8]

Entity Type	**Definition**
REACTION PRODUCT	A product is a substance that is formed during a chemical reaction.
STARTING MATERIAL	A substance that is consumed in the course of a chemical reaction providing atoms to products is considered as starting material.
REAGENT CATALYST	A reagent is a compound added to a system to cause or help with a chemical reaction. Compounds like catalysts, bases to remove protons or acids to add protons must be also annotated with this tag.
SOLVENT	A solvent is a chemical entity that dissolves a solute resulting in a solution.
OTHER COMPOUND	Oth er chemical compounds that are not the products, starting materials, reagents, catalysts and solvents.
TIME	The reaction time of the reaction.
TEMPERATURE	The temperature of the reaction.
YIELD PERCENT	Yields given in percent values.
YIELD OTHER	Yields provided in other units than %.
WORKUP	A manipulation required to isolate and purify the product of a chemical reaction
REACTION STEP	An event that converts starting materials into a product
Arg1	The elation between an event trigger word and a chemical compound
ARGM	The relation between an event trigger word and a temperature, time, or yield entity

Table 3.2: Distribution of entities across the train/test/dev split

Entity	Train	Dev	Test
REACTION STEP	3814	422	1975
EXAMPLE LABEL	888	96	471
TIME	2084	234	1149
SOLVENT	1195	130	594
WORKUP	3058	331	1645
OTHER COMPOUND	6173	705	3331
STARTING MATERIAL	2693	282	1487
REAGENT CATALYST	1917	240	968
TEMPERATURE	2623	279	1365
REACTION PRODUCT	2973	302	1624
YIELD OTHER	2088	238	1134
YIELD PERCENT	973	110	527

3.5 Evaluation

We evaluate our method using precision, recall, and F_1 scores formally defined in Equations 3.5.1 - 3.5.3. Precision is the ratio between correctly predicted mentions over the total set of predicted mentions for a specific entity, recall is the ratio of correctly predicted mentions over the actual number of mentions, and F_1 is the harmonic mean between precision and recall.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.5.1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.5.2}$$

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.5.3}$$

Where:

- TP: True Positives (correctly predicted positive instances)
- FP: False Positives (incorrectly predicted positive instances)
- FN: False Negatives (incorrectly predicted negative instances)

3.6 Experimental Details

We define the neural network architecture that we are using for the NER task. This architecture includes components from the transformers library, as well as custom layers built using PyTorch. The system is designed for NER using a few-shot learning approach facilitated by the MAML algorithm. The main components include a BERT model for embedding generation, a BiLSTM network for contextual processing, and a linear layer for entity prediction.

BERT Layer:

- Model Used: The architecture employs the "bert-base-cased" model from Hugging Face's Transformers library. This model is a pre-trained version of BERT, which is a transformer-based machine learning technique developed by Google.

- Configuration: The BERT model is utilized to convert input text into contextual embeddings.
- Parameter Freezing: After loading, all parameters of BERT are frozen. This means that during training, the weights of the BERT model will not be updated. Freezing is typically done to prevent overfitting on smaller datasets or to speed up training since only the layers added on top of BERT will be trained. However, these parameteres are not frozen for the Baseline.

Dropout Layer:

- Purpose: A dropout layer is added with a probability of 0.1 to reduce overfitting by randomly setting input units to 0 during training at each step. This helps in making the model more robust as it learns to make predictions with a reduced set of features.

BiLSTM Layer:

- Configuration: The Bi-directional Long Short-Term Memory (BiLSTM) layer is configured to take the 768-dimensional output from BERT as its input size. It has 100 units per direction, making the total output from this layer 200 units wide (100 from forward pass and 100 from backward pass). The layer uses 2 layers (number of layers=2), enhancing the model's ability to capture information from both past and future context within the data.
- Bidirectionality: The bidirectionality of the LSTM allows it to process data in both forward and backward directions, capturing patterns from all available context, which is particularly useful in sequence labeling tasks like NER.
- Purpose: This component is crucial for capturing temporal dependencies and context in data, which enhances the model's ability to understand the sequence and improve entity recognition.

Linear Layer:

- Transformation: A linear transformation is applied to the LSTM output, converting the 200-dimensional output to a 1-dimensional output per time step.

This layer is typically used to map the rich feature representation learned by the LSTM into a space where it can be converted into a probability score indicating the likelihood of a token being part of a named entity.

Sigmoid Activation:

- Output Activation: The final layer uses a sigmoid activation function to convert the logits into probabilities between 0 and 1, indicating how likely each token is to belong to a particular entity class.

Forward Pass:

- Input Handling: The forward method of the NERModel accepts two arguments: input ids and attention mask, which are standard inputs for BERT based models.
- BERT Processing: These inputs are first passed through the BERT model, where the model extracts contextual embeddings for each token.
- Dropout Application: The embeddings then pass through a dropout layer with a rate of 0.1 is applied after the BERT embeddings. This layer randomly sets input elements to zero with a frequency of 0.1 at each step during training time, which helps prevent overfitting by providing a form of regularization especially that the CHEMU dataset is not very large
- BiLSTM Processing: Subsequent to dropout, the embeddings are processed by the BiLSTM layer, which considers both past and future tokens to enhance the embeddings further.
- Linear Transformation: The BiLSTM's output is then transformed by a linear layer.
- Sigmoid Activation: Finally, a sigmoid function is applied to derive probabilities from the logits.

MetaLearner Class: This serves as a wrapper for the NERModel. It simply forwards the inputs through to the NERModel. This setup is typical in meta-learning frameworks like MAML, where multiple instances of the model may be created and manipulated during the training process.

Meta-Learner Setup: A MetaLearner class encapsulates the NERModel, allowing the meta-learning framework to handle training and adaptation processes. For the adaptation and Meta-Optimization, the model undergoes, during training, a two-level optimization process:

- Task-Specific Adaptation (Inner Loop): For each task, the model parameters are adapted using a few annotated examples from the task's support set. This step involves forward and backward passes through the NER model to adjust parameters specifically for the task.
- Meta-Optimization (Outer Loop): After adapting to individual tasks, the model's initial parameters are updated based on the performance across tasks. This step uses the meta-loss computed from the query sets to adjust the initial parameters so that the model can better adapt to new tasks with minimal training data.

Model Selection Using Minimum Loss Tracking instead of Early stopping: Instead of employing an early stopping mechanism, the algorithm utilizes a method to track and save the model with the lowest training loss across all epochs. This approach involves maintaining a record of the lowest loss observed and its corresponding epoch. The script checks after each epoch if the current training loss is lower than any previously recorded loss. If it is, the script updates the record of the minimum loss and saves the current state of the model. This method ensures that the model with the best performance on the dev set, in terms of loss minimization, is retained after the completion of all epochs.

Integration in NER: This model architecture is designed for high performance in tasks requiring deep contextual understanding, such as NER, by leveraging both pretrained embeddings and sequence modeling through LSTM. The use of a frozen BERT model as a feature extractor combined with trainable layers on top allows for efficient training on smaller datasets typical of few-shot learning scenarios.

Table 3.3 provides a complete overview of the hyperparameters that were used in our system.

Table 3.3: The MAML Parameters and Hyperparameters Used in our Implementation

Parameter	**Value**
Epochs	100
K-Query/Split	$k = 100, q = 3k$
Meta Learning Rate	0.001
N-Way	2-way
Update Learning Rate	0.1
Adaptation Learning Rate	0.1 (same as update learning rate)
Batch Size	5
Dropout Rate	0.1
LSTM Hidden Size	100
LSTM Number of Layers	2
Frozen BERT Layers	All layers frozen
BERT Model Variant	‘bert-base-uncased‘
Max Token Length	512

Chapter 4

Results

4.1 Baseline NER Results

Table 4.1 shows the precision, recall and F1 score for the REACTION STEP, REACTION PRODUCT and STARTING MATERIAL entities. The results show that the baseline system is able to extract these entity types using all of the training data with a high level of precision and recall.

Entity	Precision	Recall	F1-Score
REACTION STEP	0.94	0.98	0.96
REACTION PRODUCT	0.98	0.98	0.98
STARTING MATERIAL	0.98	0.98	0.98

Table 4.1: Baseline NER results for the CHEMU dataset.

Analysis of NER Baseline Results

Overall Performance:

- The model demonstrates excellent performance across different entity types for the CHEMU dataset. The high scores in precision, recall, and F1-score suggest that the model is both accurate and reliable in identifying and classifying the named entities.

By Entity Type:

- Reaction Step: This entity has a precision of 0.94 and a recall of 0.98, resulting in an F1-score of 0.96. The high recall indicates that the model is particularly effective at capturing most of the relevant instances for this category, while the slightly lower precision suggests a small number of false positives.
- Reaction Product and Starting Material: Both these entity types show remarkably high precision and recall scores of 0.98, leading to an F1-score of 0.98. These results indicate that the model is exceptionally proficient at identifying and correctly classifying entities as either reaction products or starting materials with minimal error.

Implications

- The high precision across all categories indicates that when the model predicts an entity, it is very likely to be correct. This is crucial for tasks where false positives can lead to misleading downstream effects, such as in automated chemical synthesis planning.
- The high recall scores show that the model misses very few actual entities, which is beneficial for comprehensive entity extraction in complex chemical data.
- Consistently high F1-scores across the entities show a balanced model that does not overly favor either precision or recall, making it robust and reliable for practical applications in chemical entity recognition.

The performance of the model on the CHEMU dataset is impressive, indicating that the combination of BERT embeddings with a BiLSTM layer effectively captures both the contextual nuances and the sequential nature of the data. This analysis shows that the Baseline model architecture and training strategy are well-suited for the NER tasks in the chemical domain. In the next section, we will see how this model performs on a MAML algorithm.

Shots	REACTION PRODUCT			REACTION STEP			STARTING MATERIAL		
	P	R	F1	P	R	F1	P	R	F1
1	0.25	0.20	0.22	0.01	0.30	0.02	0.15	0.35	0.21
2	0.30	0.25	0.27	0.01	0.35	0.02	0.18	0.40	0.25
3	0.35	0.28	0.31	0.01	0.40	0.02	0.19	0.45	0.27
4	0.37	0.29	0.33	0.01	0.45	0.02	0.20	0.48	0.28
5	0.42	0.30	0.35	0.01	0.48	0.02	0.21	0.53	0.30
10	0.39	0.35	0.37	0.01	0.50	0.02	0.21	0.50	0.30
25	0.40	0.45	0.42	0.02	0.50	0.03	0.21	0.55	0.31
50	0.42	0.48	0.45	0.02	0.53	0.04	0.21	0.58	0.32
100	0.45	0.67	0.54	0.04	0.66	0.06	0.24	0.68	0.35
200	0.46	0.68	0.55	0.05	0.66	0.08	0.25	0.69	0.36
300	0.47	0.68	0.56	0.05	0.67	0.08	0.25	0.70	0.37
400	0.47	0.69	0.56	0.05	0.67	0.08	0.25	0.70	0.37

Table 4.2: MAML Results

4.2 Few shot learning Results

Table 4.2 shows the precision (P), recall (R) and F-1 score (F1) for the REACTION PRODUCT, REACTION STEP, and STARTING MATERIAL for k-shot learning where we vary k from 1-X.

Reaction Product:

- As the number of shots increases from 1 to 50, Precision improves gradually from 0.25 to 0.42. This indicates that the model is better at correctly identifying Reaction Products with more examples.
- Recall starts at 0.20 for 1 shot and goes up to 0.48 for 50 shots, suggesting the model is capturing more true positives as it has more instances to learn from. Also, the Recall improvement rate seems to accelerate after 10 shots.
- The F1 score, which combines Precision and Recall, also shows a progressive increase, with a notable jump between 10 shots (F1=0.37) and 25 shots

(F1=0.42), indicating a significant improvement in the balanced measure of the model's performance.

Reaction Step:

- Precision remains constant at 0.01 for shots 1 through 4 and slightly increases to 0.02 for shots 25 and 50. This suggests the model's ability to correctly predict Reaction Steps is not significantly improving, possibly due to limited variability or complexity within these examples which suggest that the model needs more data to learn Reaction Step.
- Recall increases from 0.30 to 0.53. This improvement indicates the model's increasing ability to detect most of the true Reaction Step entities.
- The F1 score, being heavily influenced by the low Precision, remains low throughout but does show a small improvement.

Starting Material:

- Precision increases slightly from 0.15 to 0.21, showing a small improvement in the model's precision in identifying Starting Material entities.
- Recall sees a more substantial increase from 0.35 to 0.58 across the shots, suggesting that as the number of shots goes up, the model is becoming more comprehensive in capturing the Starting Material entities.
- The F1 score starts at 0.21 and rises to 0.32, reflecting a notable improvement in the model's performance.

4.3 Discussion

In our analysis, we observed a consistent trend of improvement across all evaluation metrics as the number of shots increases. This aligns with the fundamental principle of few-shot learning, where exposure to more examples generally leads to enhanced model performance. With each additional shot, the model gains valuable insights and refines its ability to recognize and classify entities, resulting in progressively better performance across various evaluation metrics. This trend underscores the efficacy of the few-shot learning paradigm in facilitating rapid adaptation to new tasks with minimal labeled data. However, it's important to note that while increasing shot size tends to improve performance, there may be diminishing returns beyond a certain threshold, as the model may reach a saturation point where additional examples provide diminishing marginal benefits.

The most significant gains in our analysis were observed in Recall metrics, particularly for entities such as Reaction Steps and Starting Material. This phenomenon suggests that these entities pose initial detection challenges for the model, possibly due to their nuanced contextual representations or sparse occurrence in the training data. However, as the model is exposed to more examples during training, it becomes increasingly adept at recognizing these entities, leading to significant improvements in Recall. The notable gains in Recall metrics highlight the model's capacity to learn from limited data and adapt its decision boundaries to capture the underlying patterns associated with these entities. These findings underscore the importance of robust training regimes and continual exposure to diverse examples for improving model performance, especially in scenarios with sparse annotated data.

Despite the overall improvement in model performance, Precision metrics for Reaction Step exhibit a notable discrepancy compared to other entities. This discrepancy could stem from various factors, including the inherent complexity of distinguishing Reaction Step from other entities or the lack of distinctive features in the provided examples. The relatively low Precision scores for Reaction Step suggest that the model may struggle to maintain a high level of precision while minimizing false positives, possibly due to ambiguity or overlapping characteristics with other entity types. Addressing these challenges may require further refinement of the model architecture, optimization of feature representations, or targeted training strategies tailored to improve the model's precision in identifying Reaction Steps.

The observed stabilization of the F1 score, particularly notable for Reaction Step, even with an increase in the number of shots, raises concerns about the inherent limitations of the model's current configuration. Despite improvements in individual metrics, the overall F1 score remains relatively stagnant, suggesting that the model's performance may plateau beyond a certain point. This stagnation in performance could be attributed to constraints imposed by the frozen BERT layers, which may limit the model's capacity to capture subtle nuances or complex relationships within the data. Addressing these limitations may require revisiting the model architecture, exploring alternative pre-training strategies, or incorporating additional layers of context-specific information to enhance the model's discriminative capabilities and overall performance.

4.4 Conclusion

In this work, we proposed to answer two research questions:

- Research Question 1: How do varying shot sizes impact the performance and adaptability of MAML-based NER models in handling limited data scenarios?
- Research Question #2: How transferable are the learned representations and adaptation strategies acquired by MAML-based NER models across closely related and distinct entity types within the chemical domain?

To answer Research Question 1, we conducted a comprehensive experimental study that systematically investigated the behavior of MAML-based NER models under diverse experimental conditions. We will explore the impact of varying shot sizes, ranging from very few examples per class to larger support sets, to assess the scalability and adaptability of MAML to different data regimes.

For Shot Size Variability, our analysis revealed that varying the number of examples per task (shot size) is pivotal for assessing the scalability and adaptability of MAML-based models in handling limited data scenarios. Smaller shot sizes present a challenging scenario where the model must generalize from very few examples, thereby testing its ability to learn from sparse data. Conversely, larger shot sizes provide more training instances, enabling the model to leverage additional information for task adaptation. However, despite exploring MAML models with different shot settings, our findings indicated that none outperformed the baseline due to constraints imposed by the frozen BERT architecture. This highlights the significance of considering model architecture and its impact on performance, particularly in few-shot learning scenarios.

Analyzing different task distributions allows us to assess the model's flexibility across varied contexts, such as our CHEMU dataset, within the same overarching task of NER. This exploration reveals how domain-specific knowledge and the diversity of data impact the effectiveness of meta-learning strategies. Specifically, the MAML algorithm, as illustrated in Line 1 of the MAML algorithm, considers the task distribution. In our case, we observed three entities with different task distributions, resulting in varying performance outcomes. Interestingly, one entity performed better than others, highlighting the impact of task distribution on model performance. These findings emphasize the importance of considering task distribution when designing and evaluating meta-learning approaches for NER tasks.

For Research Question #2, the consistent increase in performance metrics with larger shot sizes indicates the model's ability to effectively utilize learned representations to adapt to new examples of the same entity type. This is particularly evident in the significant improvements observed in entities like REACTION PRODUCT and STARTING MATERIAL as the number of shots increases. The model's capacity to leverage a greater number of training examples enables it to refine its learned representations and adapt more effectively to variations in context and data distribution. Consequently, we observe enhanced precision, recall, and overall F1-score, reflecting the model's improved capability to accurately identify and classify instances of these entity types.

However, the minimal improvement in precision observed for REACTION STEP suggests a limitation in the transferability of learned strategies, especially in cases where entities have less distinctive features or where the model requires more examples to capture the complexity of the entity. Despite the availability of additional training examples with larger shot sizes, the model may struggle to generalize effectively from sparse or ambiguous data, resulting in only marginal improvements in precision. This highlights the importance of considering the inherent characteristics and complexities of different entity types when evaluating the performance and adaptability of machine learning models, particularly in specialized domains like chemistry.

Overall, the observed trends in performance metrics underscore the dynamic interplay between shot size, entity complexity, and the model's adaptability, offering valuable insights into the mechanisms underlying transfer learning and few-shot learning in NER tasks within the chemical domain.

4.5 Future Work

The results of our study highlight the model's robust capability to distinguish between entity types with clear defining features. However, for closely related entities like REACTION STEP and STARTING MATERIAL, the model's generalization is noticeably constrained by the inherent complexity and similarity of these entities. This suggests a need for further refinement of adaptation strategies or the adoption of more specialized training regimes tailored to address the nuanced distinctions between closely related entities. To enhance the transferability of learned representations and adaptation strategies across such entities, future work may benefit from the integration of additional layers of contextual or feature-specific training. This could involve the exploration of different preprocessing approaches or tokenization techniques specifically designed to capture the chemical specificity inherent in these entities. Additionally, for model optimization, it is imperative to address the current limitation of frozen BERT due to GPU memory constraints, potentially by implementing strategies to unfreeze BERT during training. Furthermore, to tackle class imbalances and improve model performance, the implementation of class weighting techniques is warranted. Finally, experimenting with other hyperparameters holds promise for further enhancing the model's performance and generalization capabilities across diverse entity types within the chemical domain

Bibliography

[1] Trapit Bansal, Rishikesh Jha, and Andrew McCallum. Learning to few-shot learn across diverse natural language classification tasks. *CoRR*, abs/1911.03863, 2019.

[2] Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 522–534, Online, November 2020. Association for Computational Linguistics.

[3] Peng Chen, Jian Wang, Hongfei Lin, Di Zhao, and Zhihao Yang. Few-shot biomedical named entity recognition via knowledge-guided instance generation and prompt contrastive learning. *Bioinformatics*, 39(8):btad496, 08 2023.

[4] Cyprien de Lichy, Hadrien Glaude, and William Campbell. Meta-learning for few-shot named entity recognition. In *Proceedings of the 1st Workshop on Meta Learning and Its Applications to Natural Language Processing*, pages 44–58, Online, August 2021. Association for Computational Linguistics.

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

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *CoRR*, abs/1703.03400, 2017.

[7] Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. Few-shot classification in named entity recognition task. *CoRR*, abs/1812.06158, 2018.

[8] Jiayuan He, Dat Quoc Nguyen, Saber A Akhondi, Christian Druckenbrodt, Camilo Thorne, Ralph Hoessel, Zubair Afzal, Zenan Zhai, Biaoyan Fang, Hiyori Yoshikawa, et al. An extended overview of the clef 2020 chemu lab. In *the Conference and Labs of the Evaluation Forum (CLEF)*. 22-25 September 2020, 2020.

[9] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[10] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. 2015.

[11] Jing Li, Billy Chiu, Shanshan Feng, and Hao Wang. Few-shot named entity recognition via meta-learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(9):4245–4256, 2022.

[12] Tingting Ma, Huiqiang Jiang, Qianhui Wu, Tiejun Zhao, and Chin-Yew Lin. Decomposed meta-learning for few-shot named entity recognition. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1584–1596, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[13] Aaron Mueller, Kanika Narang, Lambert Mathias, Qifan Wang, and Hamed Firooz. Meta-training with demonstration retrieval for efficient few-shot learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6049–6064, Toronto, Canada, July 2023. Association for Computational Linguistics.

[14] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms, 2018.

[15] David Oniani, Sonish Sivarajkumar, and Yanshan Wang. Few-shot learning for clinical natural language processing using siamese neural networks, 2022.

[16] Archit Parnami and Minwoo Lee. Learning from few examples: A summary of approaches to few-shot learning, 2022.

[17] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning, 2017.

[18] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning, 2017.

[19] Yi Yang and Arzoo Katiyar. Simple and effective few-shot named entity recognition with structured nearest neighbor learning, 2020.

[20] Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. Diverse few-shot text classification with multiple metrics, 2018.

[21] Jing Zhou, Yanan Zheng, Jie Tang, Jian Li, and Zhilin Yang. Flipda: Effective and robust data augmentation for few-shot learning, 2022.

Biography

Nourchene Bargaoui

May 2024

Nourchene Bargaoui was born on March 8, 1999, and hails from Tunisia, North Africa. She pursued her undergraduate studies at The National Institute of Applied Science and Technology. Nourchene conducted multiple research stays notably at the University of Moncton in Canada and the Technical University of Dresden in Germany, reflecting her early interests in global education and engineering. In July 2022, Nourchene received her Bachelor's degree in Industrial Engineering and soon after, moved to the United States on a Fulbright scholarship from the Department of State to attend Virginia Commonwealth University (VCU) for her master's degree.

During her academic career, Nourchene has traveled extensively, both for pleasure and academic pursuits, becoming a true world traveler. Her notable achievements include receiving the DAAD Research Scholar award, an accolade that highlights her contributions to research and her academic excellence. In April 2024, she was inducted as a lifetime member of the Phi Beta Delta Society - Honor Society for International Scholars, which recognizes and encourages achievements in international education and exchange.

Currently, Nourchene is actively involved with the Fulbright Association and continues to engage in research projects and scholarly activities at VCU. Her journey reflects a continuous pursuit of knowledge and cultural exchange, shaping her into a distinguished scholar and a global citizen.