Enhancing Multilingual Information Extraction Towards Global Linguistic Inclusivity

by

Minh Nguyen

A dissertation accepted and approved in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

in Computer Science

University of Oregon

Spring 2024

DISSERTATION ABSTRACT

Minh Nguyen

Doctor of Philosophy in Computer Science

Title: Enhancing Multilingual Information Extraction Towards Global Linguistic Inclusivity

In our interconnected world, the diversity of around 7,000 languages presents challenges and opportunities for bridging language barriers. Multilingual information extraction (Multilingual IE) is crucial in natural language processing (NLP) for extracting information from texts across languages, facilitating global understanding and information equity. Despite advancements, the focus on high-resource languages has marginalized speakers of less-represented languages. Multilingual IE seeks to correct this by embracing linguistic diversity and inclusivity. This dissertation enhances Multilingual IE to address challenges of linguistic diversity, data scarcity, and model generalization, aiming to make IE technologies more accessible. It focuses on developing sophisticated algorithms for tasks like event trigger detection, event argument extraction, entity mention recognition, and relation extraction. The goal is to create a system capable of accurate information extraction across diverse languages, supporting global communication and cultural preservation. Furthermore, the importance of IE in the era of large language models (LLMs) remains significant. While LLMs have broadened NLP's capabilities, the precise, context-specific information provided by IE is essential, especially in retrieval-augmented generation (RAG) settings. This underscores IE's ongoing relevance, ensuring LLMs retrieve accurate, relevant information and highlighting IE's critical role in advancing NLP.

This dissertation includes both previously published and co-authored material.

CURRICULUM VITAE

NAME OF AUTHOR: Minh Nguyen

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon, Eugene
Hanoi University of Science and Technology, Hanoi, Vietnam

DEGREES AWARDED:

Doctor of Philosophy, Computer Science, 2024, University of Oregon
Bachelor of Engineering, Information Systems, 2019, Hanoi University of Science and Technology, Hanoi, Vietnam

AREAS OF SPECIAL INTEREST:

Natural Language Processing
Information Extraction
Multilingual Learning
Question Answering
Large Language Models

PROFESSIONAL EXPERIENCE:

Teaching Assistant, University of Oregon, 2019, 2024
Research Assistant, University of Oregon, 2020-2023
Applied Scientist Intern, Amazon Alexa AI, 2022-2024
Research Scientist Intern, Adobe Research, 2022

GRANTS, AWARDS AND HONORS:

Gurdeep Pall Graduate Student Fellowship, University of Oregon, 2022-2023
Erwin & Gertrude Juilfs Scholarship, University of Oregon, 2021
Outstanding Demo Paper Award, European Chapter of the ACL, 2021

PUBLICATIONS:

Minh Nguyen, Kishan KC, Toan Nguyen, Ankit Chadha, and Thuy Vu. (2023). Efficient Fine-tuning Large Language Models for Knowledge-Aware Response Planning. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases.*

Minh Nguyen, Kishan KC, Toan Nguyen, Thien Huu Nguyen, Ankit Chadha, and Thuy Vu. (2023). Question-Context Alignment and Answer-Context Dependencies for Effective Answer Sentence Selection. In *Proceedings of INTERSPEECH 2023.*

Minh Nguyen, Bonan Min, Franck Dernoncourt, and Thien Huu Nguyen. (2022). Learning Cross-Task Dependencies for Joint Extraction of Entities, Events, Event Arguments, and Relations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing.*

Minh Nguyen, Bonan Min, Franck Dernoncourt, and Thien Huu Nguyen. (2022). Joint Extraction of Entities, Relations, and Events via Modeling Inter-Instance and Inter-Label Dependencies. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*

Minh Nguyen, Nghia Trung Ngo, Bonan Min, and Thien Huu Nguyen. (2022). FAMIE: A Fast Active Learning Framework for Multilingual Information Extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations.*

Minh Nguyen, Franck Dernoncourt, and Thien Huu Nguyen. (2022). BehanceMT: A Machine Translation Corpus for Livestreaming Video Transcripts. In *Proceedings of the First Workshop On Transcript Understanding.*

Minh Nguyen, Tuan Ngo Nguyen, Bonan Min and Thien Huu Nguyen. (2021). Crosslingual Transfer Learning for Relation and Event Extraction via Word Category and Class Alignments. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing.*

Minh Nguyen, Viet Dac Lai and Thien Huu Nguyen. (2021). Cross-Task Instance Representation Interactions and Label Dependencies for Joint Information Extraction with Graph Convolutional Networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*

Minh Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh and Thien Huu Nguyen. (2021). Trankit: A Light-Weight Transformer-based Toolkit for Multilingual Natural Language Processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations.*

Minh Nguyen and Thien Huu Nguyen. (2021). Improving Cross-Lingual Transfer for Event Argument Extraction with Language-Universal Sentence Structures. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop.*

ACKNOWLEDGEMENTS

First and foremost, I extend my deepest gratitude to my advisor, Professor Thien Huu Nguyen, whose guidance, support, and insightful critiques have been indispensable throughout this journey. Your unwavering faith in my capabilities and your dedication to my academic and personal growth have been truly inspirational. I am immensely grateful for your mentorship and patience.

I am also sincerely grateful to Dr. Bonan Min, who led the IARPA Better Extraction from Text Towards Enhanced Retrieval (BETTER) project, where many ideas in this dissertation were developed while I worked as a research assistant to meet some of the project's demands. Thank you for your leadership and the valuable skills I have learned throughout this project. Your guidance and the opportunities you provided have greatly contributed to my growth as a researcher.

I would also like to express my heartfelt thanks to the members of my dissertation committee and the dissertation advisory committee: Professor Thanh Hong Nguyen, Professor Daniel Lowd, Professor Kristopher Kyle, and Professor Humphrey Shi. Your rigorous feedback, valuable insights, and constructive suggestions have significantly contributed to the depth and breadth of my research. Your expertise and dedication to excellence have been instrumental in shaping my scholarly work.

To my labmates and friends, Viet Dac Lai, Amir Pouran Ben Veyseh, Qiuhao Lu, Luis Fernando Guzman Nateras, Zayd Hammoudeh, Tuan Ngo Nguyen, Nghia Trung Ngo, Hieu Man Duc Trong, and Chien Van Nguyen, thank you for the camaraderie, intellectual exchanges, and countless hours of discussion that have enriched my research experience. Your support, both academically and personally, has been a constant source of encouragement. I am fortunate to have been part of such a collaborative and inspiring team.

I am also grateful to the faculty and administrative personnel of the Department of Computer Science, Professor Hank Childs, Cheri Smith, and Nicole Moynahan. Your commitment to creating an enriching academic environment and your support in navigating the complexities of graduate study have been invaluable. Thank you for your assistance, encouragement, and for making the Department of Computer Science a welcoming place for learning and research.

My experience at Amazon was profoundly enriching, and I owe a great deal of gratitude to my internship managers and mentors, Thuy Vu, Toan Quoc Nguyen, Kishan KC, and Zeyu Zhang. Your guidance, expertise, and willingness to share your knowledge have greatly contributed to my professional development. Thank you for providing me with this opportunity and for your supportive and inspiring leadership.

Finally, my deepest appreciation goes to my family: my mother, my father, and my wife. Your unconditional love, unwavering support, and constant encouragement have been my strength. Thank you for believing in me, for your sacrifices, and for being my source of motivation and comfort throughout this journey. This dissertation is not just my achievement, but a testament to the enduring support and love you have provided me.

To everyone who has been a part of my journey, I extend my sincerest thanks. Your contributions have shaped my academic path and personal growth in ways that words cannot fully express.

To my beloved family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table Page

Table Page

CHAPTER I

INTRODUCTION

The majority of this chapter's content is derived from my dissertation proposal, where I served as the primary author, while Thien Huu Nguyen contributed through editorial recommendations.

1.1 Overview

In our modern world, language plays a crucial role in shaping cultures and identities. With about 7,000 languages spoken worldwide, each carrying its unique expressions and meanings, we face a significant challenge in the field of information technology, especially in communication and information access (Joshi, Santy, Budhiraja, Bali, & Choudhury, 2020; Zaugg, 2020). As global interaction intensifies, the demand for technologies that can overcome language barriers has reached new heights. Among these technologies, multilingual information extraction (Multilingual IE), a field within natural language processing (NLP), stands out as a key player (Névéol et al., 2017; Poibeau, Saggion, Piskorski, & Yangarber, 2013; Pouran Ben Veyseh, Nguyen, Dernoncourt, & Nguyen, 2022; Ro, Lee, & Kang, 2020).

Multilingual IE is a vital area within NLP tasked with extracting structured information from unstructured text across a variety of languages (Y. Lin, Ji, Huang, & Wu, 2020b; Luan et al., 2019b; M. V. Nguyen, Lai, & Nguyen, 2021; M. V. Nguyen, Min, Dernoncourt, & Nguyen, 2022a, 2022b). This capability is essential; in a time when information equates to power, being able to understand and process information across languages is invaluable. This is not just about technology but about bridging gaps in understanding, facilitating cultural exchanges, and making knowledge accessible to all. However, reaching these goals is filled with challenges, complexities, and nuances that require a thorough exploration of the field's current state, its obstacles, and the potential solutions it offers (V. Lai, Man, Ngo, Dernoncourt, & Nguyen, 2022; V. D. Lai, Veyseh, Nguyen, Dernoncourt, & Nguyen, 2022; Pouran Ben Veyseh, Ebrahimi, Dernoncourt, & Nguyen, 2022).

Traditionally, the bulk of NLP research has focused on a few high-resource languages, with English being the primary focus (Hovy & Prabhumoye, 2021; Søgaard, 2022). This concentration on a select few languages leaves speakers of less-resourced languages at a disadvantage, missing out on the full benefits that NLP technologies can provide (Adelani et al., 2021). This imbalance not only limits global communication and information access but also contributes to inequality in knowledge distribution and technological progress (Zaugg, 2020). The development of Multilingual IE is a critical step towards addressing these issues. By aiming to process text in a wide range of languages, Multilingual IE strives to ensure no language community is overlooked in the digital era.

At the core of Multilingual IE are several interconnected challenges that reflect the complexity of human language. Languages differ in their vocabulary, grammar, meaning conveyance, information structure, and world conceptualization (Blommaert, 2013; Evans, 2018; Giunchiglia, Batsuren, Bella, et al., 2017; Pacheco Coelho et al., 2019). These differences pose significant challenges to creating algorithms and models that can accurately extract information across languages. The lack of digital resources and annotated datasets for many languages further complicates the ability to train models with high precision and accuracy (V. Lai et al., 2022; V. D. Lai et al., 2022; Pouran Ben Veyseh, Ebrahimi, et al., 2022; Pouran Ben Veyseh, Nguyen, et al., 2022). Despite these challenges, the field of Multilingual IE has made considerable progress (Y. Lin et al., 2020b; Minh Tran, Phung, & Nguyen, 2021; Pouran Ben Veyseh, Dernoncourt, Dou, & Nguyen, 2020), driven by several factors. For example, multilingual transformer-based language models have significantly improved text processing and understanding, laying the groundwork for multilingual capabilities (Conneau et al., 2019; Devlin, Chang, Lee, & Toutanova, 2019b). Additionally, advances in active learning and cross-lingual model training have started to mitigate the issue of data scarcity, enabling efficient development of IE models for low-resource languages (X. Chen, Awadallah, Hassan, Wang, & Cardie, 2019; Huang, Ji, & May, 2019; Lange, Iurshina, Adel, & Strötgen, 2020b; Shelmanov et al., 2021).

This dissertation is set against this backdrop in NLP and Multilingual IE research. It aims to contribute to Multilingual IE by tackling the main challenges of linguistic diversity, data scarcity, and model generalizability. By focusing on improving upstream models, developing language-agnostic downstream architectures, and advancing cross-lingual transfer learning and active learning for IE, this work seeks to extend the boundaries of Multilingual IE. In doing so, the dissertation not only aims to push forward technical advancements but also to contribute to a more inclusive, equitable, and linguistically diverse digital future. Moreover, the dissertation underscores the potential of IE in the evolution of large language models (LLMs) (Achiam et al., 2023; Brown et al., 2020; Chowdhery et al., 2023; Chung et al., 2022) by introducing a novel retrieval-augmented generation (RAG) framework, where IE has a pivotal contribution to improving the retrieval system that ensures LLMs can offer accurate and reliable responses to user queries.

In conclusion, as we navigate the challenges and opportunities of technological advancement and global linguistic diversity, the role of Multilingual

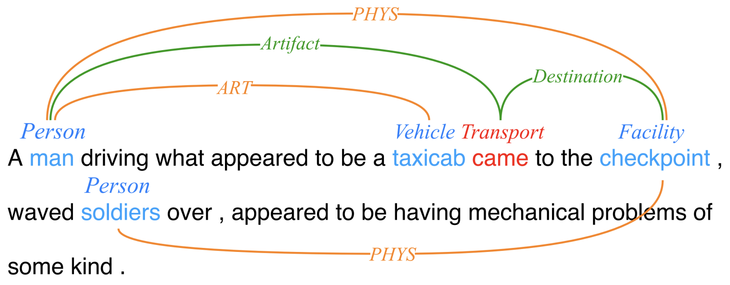

PHYS
Artifact
Destination
ART
Person
Vehicle
Transport
Facility
A man driving what appeared to be a taxicab came to the checkpoint ,
Person
waved soldiers over , appeared to be having mechanical problems of
PHYS
some kind .

Figure 1. An example with annotations for four main IE tasks: event trigger detection, event argument extraction, entity mention recognition, and relation extraction (M. V. Nguyen, Lai, & Nguyen, 2021).

IE has become increasingly important. This dissertation recognizes the complexity of the tasks ahead but is driven by the potential impact that advancements in this area could have on global communication, information accessibility, and cultural preservation. Through dedicated research, innovative approaches, and a commitment to inclusivity, this work intends to play a part in the ongoing development of NLP technologies, ensuring they serve a wide and diverse global audience.

1.2 Problem Definitions

The pivotal role of multilingual information extraction (Multilingual IE) is underscored by the challenge of interpreting and structuring the vast and varied information embedded in text. The complexity of the field is multiplied when considering the diversity of the world's languages and the nuances inherent in each. To automate the understanding and extraction of information across languages, Multilingual IE encompasses several key tasks, each with its unique challenges and methodologies. These tasks include event trigger detection (ETD), event argument extraction (EAE), entity mention recognition (EMR), and relation extraction (RE). Figure 1 illustrates a sentence annotated with these tasks, showcasing the intricate interplay between different elements within a text.

- **Event trigger detection (ETD)**: involves identifying words or phrases that signal the occurrence of an event. In the figure, the word “came” serves as a trigger for the transportation event, indicating the action of moving towards a destination. The ability to detect such triggers is fundamental to understanding the dynamics within a text, as it sets the stage for further extraction tasks. The challenge lies in accurately pinpointing these triggers across different contexts and linguistic structures, where the same word may not always signify the same event type in every instance.
- **Event argument extraction (EAE)**: is the process of identifying and classifying the entities associated with an event trigger. Once an event trigger is detected, EAE seeks to determine the participants, objects, and attributes related to that event. For example, the man driving and the checkpoint in the provided figure are arguments related to the transportation event triggered by “came”. The difficulty in EAE is two-fold: correctly associating entities with the correct event and correctly classifying their roles, which can vary widely across languages and contexts.
- **Entity mention recognition (EMR)**: focuses on identifying and categorizing entities (persons, organizations, locations, etc.) within a text. In the sentence from the figure, “a man” and “soldiers” are recognized as persons, “taxicab” as a vehicle, and “checkpoint” as a facility. EMR is a foundational task in NLP, as it allows systems to distinguish and categorize the key components of information. The challenge with EMR, especially in a multilingual context, is dealing with the vast array of entity types and the subtleties of their mention, which can be heavily influenced by cultural and linguistic factors.

– **Relation extraction (RE)**: involves identifying the relationships between entities within a text. The figure shows several relationships: between the man and the taxicab (the man is driving the taxicab), between the man and the checkpoint (the man is moving towards the checkpoint), and between the soldiers and the checkpoints (the soldiers are physically at the checkpoint). RE is crucial for building a comprehensive picture of the interactions and connections between entities, allowing for a deeper understanding of the text. The primary challenge in RE is the complexity of relationships that can exist and the subtlety with which they can be expressed, particularly in texts with intricate sentence structures or in languages with less rigid syntax.

To address these challenges within Multilingual IE, we need models that are not only robust and scalable but also nuanced and adaptable to the wide range of linguistic cues and subtleties found across different languages. This involves developing sophisticated algorithms that can handle the ambiguity and variability of natural language, while also being sensitive to the cultural and contextual elements that influence meaning. The overarching problem this dissertation will tackle is developing a system that can integrate these tasks into a coherent framework capable of accurately performing Multilingual IE across diverse linguistic landscapes.

1.3 Research Questions

This dissertation is anchored on a set of research questions that aim to address the intricacies and challenges of Multilingual IE. The forthcoming chapters of the dissertation will delve into each question in detail, offering a thorough exploration of our proposed methods and their implications for Multilingual IE. In particular, the research questions that we would like to answer are:

- **RQ1**: How can upstream models in Multilingual IE be enhanced to improve linguistic feature extraction across languages?
- **RQ2**: What architecture can be developed for downstream IE models to be effectively language-agnostic?
- **RQ3**: Given target languages with limited or no training data, how can we build effective IE models?

The first question investigates the improvement of upstream processes that form the foundation for accurate downstream information extraction, such as sentence segmentation and part-of-speech tagging. The second question seeks to establish a robust framework for downstream models that remain effective regardless of the language input for the four main tasks of IE. The third will explore methodologies for training IE models for low-resource languages through either cross-lingual transfer learning or active learning. Furthermore, we would like to explore the question:

- **RQ4**: What is the role of IE in recent advancements of LLMs?

The final question aims to identify how IE can be employed to enhance LLMs' ability to provide accurate and reliable responses. Each of these questions will be meticulously addressed in the dissertation, with dedicated chapters that provide an in-depth analysis and discussion. These chapters will collectively form a comprehensive approach to tackling the multifaceted challenges of Multilingual IE, with the goal of contributing valuable knowledge and innovative solutions to the field.

1.4 Dissertation Outline

In the exploration of Multilingual IE, this dissertation delineates a comprehensive approach across four distinct research directions (RDs) toward answering the four research questions (RQ1, RQ2, RQ3, and RQ4) respectively. Each direction targets a specific aspect of Multilingual IE, aiming to collectively enhance the field's capability and application across multiple languages:

1.4.1 RD1: Advancements in Linguistic Feature Processing for Multilingual IE. The first direction delves into enhancing upstream models that process fundamental linguistic features such as sentence boundaries, word tags, and dependency trees, crucial for the performance of downstream IE models on the four IE tasks (see Figure 1). Previous work such as those of Manning et al. (2014) and Straka, Hajič, and Straková (2016) provide a foundation for understanding these models.

To improve upstream models in terms of speed, performance, and linguistic diversity, we propose Trankit (M. V. Nguyen, Lai, Pouran Ben Veyseh, & Nguyen, 2021), a novel transformer-based toolkit designed for multilingual NLP. Trankit provides a trainable NLP pipeline across over 100 languages, alongside 90 pretrained pipelines covering 56 languages. Anchored by a state-of-the-art pretrained language model (Conneau et al., 2019), Trankit surpasses existing multilingual NLP pipelines in performance across several key tasks, including sentence segmentation, part-of-speech tagging, morphological feature tagging, and dependency parsing. Despite incorporating a large pretrained transformer model, Trankit maintains efficiency in terms of memory use and processing speed. This efficiency is achieved through a novel plug-and-play mechanism featuring Adapters (Pfeiffer, Vulić, Gurevych, & Ruder, 2020), allowing for a single multilingual pretrained transformer to be utilized across different language pipelines. Details of Trankit are presented in chapter II.

1.4.2 RD2: Language-Agnostic Models for Joint Information Extraction. Shifting the focus from linguistic feature processing to the architecture of IE models themselves, this direction aims to develop models that can be universally applied across languages without requiring language-specific modifications. This includes a comparative analysis of traditional pipelined approaches (T. H. Nguyen & Grishman, 2015b; G. Zhou, Su, Zhang, & Zhang, 2005b) against joint models, known as Joint Information Extraction (JointIE), which perform a suite of IE tasks within a single model architecture. The comparative study assesses how these models manage error propagation and leverage the interdependencies between tasks, with references to the works of Luan et al. (2019b), Y. Lin et al. (2020b), and Zhang and Ji (2021b). The development and testing of new language-agnostic models are integral to this direction, with a focus on models that minimize the need for language-specific adjustments. Furthermore, enhancing the models' ability to generalize across languages is crucial, emphasizing the importance of leveraging language differences and similarities for improved multilingual training and performance.

In this direction, chapter III introduces FourIE (M. V. Nguyen, Lai, & Nguyen, 2021), our novel model developed to tackle the four tasks of IE within a unified framework. Unlike previous efforts that have attempted to jointly address these four IE tasks (Y. Lin et al., 2020b; Luan et al., 2019b; Zhang & Ji, 2021b), FourIE stands out by offering two innovative contributions designed to capture the interdependencies between tasks effectively. The first contribution is at the representation level, where we introduce an interaction graph that connects instances across the four tasks. This graph is utilized to enhance the prediction representation of one instance by incorporating insights from related instances of the other tasks. The second contribution is at the label level, where we present a dependency graph specifically for the information types involved in the four IE tasks. This graph delineates the relationships between the types found within an input sentence, thereby capturing the intricate connections among them. Following this, we propose other innovative models that can jointly perform the four IE tasks, namely, DepIE, and GraphIE that offer more advanced mechanisms to capture such cross-task dependencies better.

1.4.3 RD3: Learning Methods for IE in Low-Resource Languages. The third direction addresses the challenge of non-existent or limited training data in target languages. In case the training data in target languages do not exist, previous work tackles this by using multilingual word embeddings (X. Chen & Cardie, 2018; Heyman, Verreet, Vulić, & Moens, 2019) and pre-trained language models (Conneau et al., 2019; Devlin et al., 2019b) to generate crosslingual representation vectors, examining their efficacy in adapting knowledge from high-resource languages to improve IE in the target languages (J. Liu, Chen, Liu, & Zhao, 2019a; M'hamdi, Freedman, & May, 2019; Subburathinam et al., 2019). Moreover, addressing monolingual bias becomes a pivotal aspect of this research direction, employing strategies such as language adversarial training to combat biases originating from the predominance of source language data in model training (X. Chen et al., 2019; Huang et al., 2019; Lange et al., 2020b). In the other case where limited training data in target languages is available, active learning can be employed to effectively annotate more training examples for maximizing the performance of the model in the target languages (Shen, Yun, Lipton, Kronrod, & Anandkumar, 2017b).

To deal with the first scenario, chapter IV presents our novel learning method called CCCAR for class- and word category-based crosslingual alignment of representations (M. V. Nguyen, Nguyen, Min, & Nguyen, 2021). Our main idea behind is to ensure similar representations of the same concepts (i.e., word categories and class labels) across source and target languages for improving the cross-lingual transferability of the model. If the training data for the target languages is limitedly available, we offer our novel active learning framework called FAMIE (M. V. Nguyen, Ngo, Min, & Nguyen, 2022). The framework employs a small proxy model for fast training and data selection, effectively building IE models for target languages through iterative annotations of more training examples.

1.4.4 RD4: Potential Applications of Information Extraction for Enhancing Large Language Models.

Recent research (Achiam et al., 2023; Brown et al., 2020; Chowdhery et al., 2023; Chung et al., 2022) highlights the importance of large language models (LLMs) in the field of NLP, owing to their exceptional capabilities across different tasks. While these LLMs have acquired a degree of world knowledge through their training process (Petroni et al., 2019; Roberts, Raffel, & Shazeer, 2020a), they are prone to generating false or imaginary information (Maynez, Narayan, Bohnet, & McDonald, 2020; C. Zhou et al., 2021a). To mitigate this issue, enhancing LLMs with the capability to retrieve accurate information from external databases has been identified as a promising approach (Izacard et al., 2022; Khandelwal, Levy, Jurafsky, Zettlemoyer, & Lewis, 2020). This method suggests that the effectiveness of LLMs could significantly rely on the quality of the data retrieved. IE has proven to be an invaluable asset in refining these retrieval processes by converting unstructured text into structured data, thereby facilitating the development of more sophisticated retrieval systems (Borisov, Aliannejadi, & Crestani, 2021; Corcoglioniti, Dragoni, Rospocher, & Aprosio, 2016) that ultimately benefit retrieval-augmented generation (RAG)-based LLMs.

In light of this, we introduce an innovative RAG framework - KARP (M. Nguyen, C, Nguyen, Chadha, & Vu, 2023) in chapter V, comprising a novel knowledge retrieval component and a LLM for open domain question answering. Given a user question, our framework employs the knowledge retriever to extract relevant words from each potential web context to assess their relevance and determine the most suitable contexts for the LLM to generate answers. Furthermore, we propose a novel finetuning method for training the LLM to efficiently exploit both external and internal knowledge for answer generation.

This dissertation contains materials from published and co-authored papers. We acknowledge all the co-authors: Thien Huu Nguyen, Amir Pouran Ben Veyseh, Viet Dac Lai, Bonan Min, Tuan Ngo Nguyen, Nghia Trung Ngo, Franck Dernoncourt, Toan Quoc Nguyen, Kishan KC, Ankit Chadha, and Thuy Vu.

CHAPTER II

ADVANCEMENTS IN LINGUISTIC FEATURE PROCESSING FOR MULTILINGUAL IE

This chapter contains materials from the published paper *"Minh Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh, and Thien Huu Nguyen.* ***'Trankit: A Light-Weight Transformer-based Toolkit for Multilingual Natural Language Processing'*** *In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, 2021"* (M. V. Nguyen, Lai, Pouran Ben Veyseh, & Nguyen, 2021). Minh was responsible for the system design and implementation, experiments, evaluation and writing as the first author. Thien, Viet and Amir provided meaningful discussion and analysis. Thien provided editorial writing for the paper submission. The paper was revised to comply with the dissertation format and purposes.

In the exploration of Multilingual IE, this dissertation delineates a comprehensive approach across four distinct research directions (RDs) toward answering the four research questions (RQ1, RQ2, RQ3, and RQ4) stated in chapter I. The first direction (RD1) delves into enhancing upstream models that process fundamental linguistic features such as sentence boundaries, word tags, and dependency trees, crucial for the performance of downstream IE models on the four IE tasks.

To improve upstream models in terms of speed, performance, and linguistic diversity, this chapter introduces Trankit, a novel transformer-based toolkit designed for multilingual NLP. Trankit provides a trainable NLP pipeline across over 100 languages, alongside 90 pretrained pipelines covering 56 languages. Anchored by a state-of-the-art pretrained language model, Trankit surpasses existing multilingual NLP pipelines in performance across several key tasks, including sentence segmentation, part-of-speech tagging, morphological feature tagging, and dependency parsing. Despite incorporating a large pretrained transformer model, Trankit maintains efficiency in terms of memory use and processing speed. This efficiency is achieved through a novel plug-and-play mechanism featuring Adapters, allowing for a single multilingual pretrained transformer to be utilized across different language pipelines.

2.1 Introduction

Many efforts have been devoted to developing multilingual NLP systems to overcome language barriers (Aharoni, Johnson, & Firat, 2019; Kanayama & Iwamoto, 2020; J. Liu, Chen, Liu, & Zhao, 2019b; M. V. Nguyen & Nguyen, 2021a; Taghizadeh & Faili, 2020; Zhu, 2020). A large portion of existing multilingual systems has focused on downstream NLP tasks that critically depend on upstream linguistic features, ranging from basic information such as token and sentence boundaries for raw text to more sophisticated structures such as part-of-speech tags, morphological features, and dependency trees of sentences (called fundamental NLP tasks). As such, building effective multilingual systems/pipelines for fundamental upstream NLP tasks to produce such information has the potentials to transform multilingual downstream systems.

There have been several NLP toolkits that concerns multilingualism for fundamental NLP tasks, featuring spaCy[1], UDify (Kondratyuk & Straka, 2019), Flair (Akbik et al., 2019), CoreNLP (Manning et al., 2014), UDPipe (Straka, 2018b), and Stanza (Qi, Zhang, Zhang, Bolton, & Manning, 2020b). However, these toolkits have their own limitations. spaCy is designed to focus on speed, thus

[1]`https://spacy.io/`

it needs to sacrifice the performance. UDify and Flair cannot process raw text as they depend on external tokenizers. CoreNLP supports raw text, but it does not offer state-of-the-art performance. UDPipe and Stanza are the recent toolkits that leverage word embeddings, i.e., word2vec (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) and fastText (Bojanowski, Grave, Joulin, & Mikolov, 2017), to deliver current state-of-the-art performance for many languages. However, Stanza and UDPipe's pipelines for different languages are trained separately and do not share any component, especially the embedding layers that account for most of the model size. This makes their memory usage grow aggressively as pipelines for more languages are simultaneously needed and loaded into the memory (e.g., for language learning apps). Most importantly, none of such toolkits have explored contextualized embeddings from pretrained transformer-based language models that have the potentials to significantly improve the performance of the NLP tasks, as demonstrated in many prior works (Conneau et al., 2020; Devlin et al., 2019b; Y. Liu et al., 2019).

In this paper, we introduce **Trankit**, a multilingual Transformer-based NLP Toolkit that overcomes such limitations. Our toolkit can process raw text for fundamental NLP tasks, supporting 56 languages with 90 pre-trained pipelines on 90 treebanks of the Universal Dependency v2.5 (Zeman et al., 2019). By utilizing the state-of-the-art multilingual pretrained transformer *XLM-Roberta* (Conneau et al., 2020), Trankit advances state-of-the-art performance for sentence segmentation, part-of-speech (POS) tagging, morphological feature tagging, and dependency parsing while achieving competitive or better performance for tokenization, multi-word token expansion, and lemmatization over the 90 treebanks. It also obtains

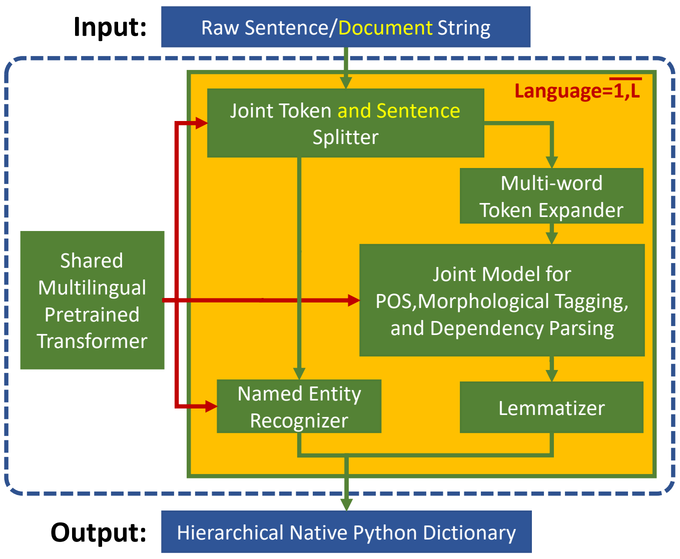

Input:
Raw Sentence/Document String
Language=1,L
Joint Token and Sentence Splitter
Multi-word Token Expander
Shared Multilingual Pretrained Transformer
Joint Model for POS,Morphological Tagging, and Dependency Parsing
Named Entity Recognizer
Lemmatizer
Output:
Hierarchical Native Python Dictionary

Figure 2. Overall architecture of Trankit. A single multilingual pretrained transformer is shared across three components (pointed by the red arrows) of the pipeline for different languages.

competitive or better performance for named entity recognition (NER) on 11 public datasets.

Unlike previous work, *our token and sentence splitter is wordpiece-based* instead of character-based to better exploit contextual information, which are beneficial in many languages. Considering the following sentence:

> *"John Donovan from Argghhh! has put out a excellent slide show on what was actually found and fought for in Fallujah."*

As such, Trankit correctly recognizes this as a single sentence while character-based sentence splitters of Stanza and UDPipe are easily fooled by the exclamation mark "!", treating it as two separate sentences. To our knowledge, this is the first work to successfully build a wordpiece-based token and sentence splitter that works well for 56 languages.

Figure 2 presents the overall architecture of Trankit pipeline that features three novel transformer-based components for: (i) the joint token and sentence splitter, (ii) the joint model for POS tagging, morphological tagging, dependency parsing, and (iii) the named entity recognizer. One potential concern for our use of a large pretrained transformer model (i.e., *XML-Roberta*) in Trankit involves GPU memory where different transformer-based components in the pipeline for one or multiple languages must be simultaneously loaded into the memory to serve multilingual tasks. This could extensively consume the memory if different versions of the large pre-trained transformer (finetuned for each component) are employed in the pipeline. As such, we introduce a novel plug-and-play mechanism with Adapters to address this memory issue. Adapters are small networks injected inside all layers of the pretrained transformer model that have shown their effectiveness as a light-weight alternative for the traditional finetuning of pretrained transformers (Houlsby et al., 2019; Peters, Ruder, & Smith, 2019b; Pfeiffer, Rücklé, et al., 2020; Pfeiffer, Vulić, et al., 2020). In Trankit, a set of adapters (for transfomer layers) and task-specific weights (for final predictions) are created for each transformer-based component for each language while only one single large multilingual pretrained transformer is shared across components and languages. Adapters allow us to learn language-specific features for tasks. During training, the shared pretrained transformer is fixed while only the adapters and task-specific weights are updated. At inference time, depending on the language of the input text and the current active component, the corresponding trained adapter and task-specific weights are activated and plugged into the pipeline to process the input. This mechanism not only solves the memory problem but also substantially reduces the training time.

2.2 Related Work

There have been works using pre-trained transformers to build models for character-based word segmentation for Chinese (Che, Feng, Qin, & Liu, 2020; Tian et al., 2020; H. Yang, 2019); POS tagging for Dutch, English, Chinese, and Vietnamese (Che et al., 2020; de Vries et al., 2019; D. Q. Nguyen & Tuan Nguyen, 2020; Tenney, Das, & Pavlick, 2019; Tian et al., 2020); morphological feature tagging for Estonian and Persian (Kittask, Milintsevich, & Sirts, 2020; Mohseni & Tebbifakhr, 2019); and dependency parsing for English and Chinese (Che et al., 2020; Tenney et al., 2019). However, all of these works are only developed for some specific language, thus potentially unable to support and scale to the multilingual setting.

Some works have designed multilingual transformer-based systems via multilingual training on the combined data of different languages (Kondratyuk & Straka, 2019; Tsai et al., 2019; Üstün, Bisazza, Bouma, & van Noord, 2020). However, multilingual training is suboptimal (see Section 2.5). Also, these systems still rely on external resources to perform tokenization and sentence segmentation, thus unable to consume raw text. To our knowedge, this is the first work to successfully build a multilingual transformer-based NLP toolkit where different transformer-based models for many languages can be simultaneously loaded into GPU memory and process raw text inputs of different languages.

2.3 Design and Architecture

Adapters. Adapters play a critical role in making Trankit memory- and time-efficient for training and inference. Figure 3 shows the architecture and the location of an adapter inside a layer of transformer. We use the adapter architecture proposed by (Pfeiffer, Rücklé, et al., 2020; Pfeiffer, Vulić, et al., 2020), which

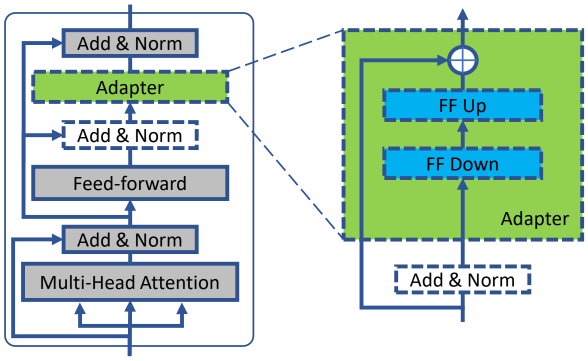


Figure 3. **Left**: location of an adapter (green box) inside a layer of the pretrained transformer. Gray boxes represent the original components of a transformer layer. **Right**: the network architecture of an adapter.

consists of two projection layers Up and Down (feed-forward networks), and a residual connection.

$$c_i = \text{AddNorm}(r_i), h_i = \text{Up}(\text{ReLU}(\text{Down}(c_i))) + r_i \tag{2.1}$$

where r_i is the input vector from the transformer layer for the adapter and h_i is the output vector for the transformer layer i. During training, all the weights of the pretrained transformer (i.e., gray boxes) are fixed and only the adapter weights of two projection layers and the task-specific weights outside the transformer (for final predictions) are updated. As demonstrated in Figure 2, Trankit involves six components described as follows.

Multilingual Encoder with Adapters. This is our core component that is shared across different transformer-based components for different languages of the system. Given an input raw text s, we first split it into substrings by spaces. Afterward, Sentence Piece, a multilingual subword tokenizer (Kudo, 2018; Kudo & Richardson, 2018), is used to further split each substring into wordpieces. By concatenating wordpiece sequences for substrings, we obtain an overall sequence of wordpieces $\mathbf{w} = [w_1, w_2, \ldots, w_K]$ for s. In the next step, $\mathbf{w}$ is fed into the pretrained transformer, which is already integrated with adapters, to obtain the wordpiece representations:

$$x_{1:K}^{l,m} = \text{Transformer}(w_{1:K}; \theta_{AD}^{l,m}) \tag{2.2}$$

Here, $\theta_{AD}^{l,m}$ represents the adapter weights for language l and component m of the system. As such, we have specific adapters in all transformer layers for each component m and language l. Note that if K is larger than the maximum input length of the pretrained transformer (i.e., 512), we further divide $\mathbf{w}$ into consecutive chunks; each has the length less than or equal to the maximum length. The pretrained transformer is then applied over each chunk to obtain a representation vector for each wordpiece in $\mathbf{w}$. Finally, $x_{1:K}^{l,m}$ will be sent to component m to perform the corresponding task.

Joint Token and Sentence Splitter. Given the wordpiece representations $x_{1:K}^{l,m}$ for this component, each vector $x_i^{l,m}$ for $w_i \in \mathbf{w}$ will be consumed by a feed-forward network with softmax in the end to predict if w_i is the end of a single-word token, the end of a multi-word token, or the end of a sentence. The predictions for all wordpieces in $\mathbf{w}$ will then be aggregated to determine token, multi-word token, and sentence boundaries for s.

Multi-word Token Expander. This component is responsible for expanding each detected multi-word token (MWT) into multiple syntactic words[2]. We follow Stanza to deploy a character-based seq2seq model for this component. This decision is made based on our observation that the task is done best at character level, and the character-based model (with character embeddings) is very small.

[2]For languages (e.g., English, Chinese) that do not require MWT expansion, tokens and words are the same concepts.

Treebank	System	Tokens	Sents.	Words	UPOS	XPOS	UFeats	Lemmas	UAS	LAS
Overall (90 treebanks)	**Trankit**	99.23	**91.82**	**99.02**	**95.65**	**94.05**	**93.21**	**94.27**	**87.06**	**83.69**
	Stanza	**99.26**	88.58	98.90	94.21	92.50	91.75	94.15	83.06	78.68
Arabic-PADT	**Trankit**	99.93	**96.59**	**99.22**	**96.31**	**94.08**	**94.28**	**94.65**	**88.39**	**84.68**
	Stanza	**99.98**	80.43	97.88	94.89	91.75	91.86	93.27	83.27	79.33
	UDPipe	99.98	82.09	94.58	90.36	84.00	84.16	88.46	72.67	68.14
Chinese-GSD	**Trankit**	**97.01**	**99.7**	**97.01**	**94.21**	**94.02**	**96.59**	**97.01**	**85.19**	**82.54**
	Stanza	92.83	98.80	92.83	89.12	88.93	92.11	92.83	72.88	69.82
	UDPipe	90.27	99.10	90.27	84.13	84.04	89.05	90.26	61.60	57.81
English-EWT	**Trankit**	98.48	**88.35**	98.48	**95.95**	**95.71**	**96.26**	96.84	**90.14**	**87.96**
	Stanza	**99.01**	81.13	**99.01**	95.40	95.12	96.11	**97.21**	86.22	83.59
	UDPipe	98.90	77.40	98.90	93.26	92.75	94.23	95.45	80.22	77.03
	spaCy	97.44	63.16	97.44	86.99	91.05	-	87.16	55.38	37.03
French-GSD	**Trankit**	**99.7**	**96.63**	**99.66**	**97.85**	-	**97.16**	**97.80**	**94.00**	**92.34**
	Stanza	99.68	94.92	99.48	97.30	-	96.72	97.64	91.38	89.05
	UDPipe	99.68	93.59	98.81	95.85	-	95.55	96.61	87.14	84.26
	spaCy	99.02	89.73	94.81	89.67	-	-	88.55	75.22	66.93
Spanish-Ancora	**Trankit**	99.94	**99.13**	99.93	**99.02**	**98.94**	**98.8**	**99.17**	**94.11**	**92.41**
	Stanza	**99.98**	99.07	**99.98**	98.78	98.67	98.59	99.19	92.21	90.01
	UDPipe	99.97	98.32	99.95	98.32	98.13	98.13	98.48	88.22	85.10
	spaCy	99.95	97.54	99.43	93.43	-	-	80.02	89.35	83.81

Table 1. Systems' performance on test sets of the Universal Dependencies v2.5 treebanks. Performance for Stanza, UDPipe, and spaCy is obtained using their public pretrained models. The overall performance for Trankit and Stanza is computed as the macro-averaged F1 over 90 treebanks. Detailed performance of Trankit for 90 supported treebanks can be found at our documentation page.

Joint Model for POS Tagging, Morphological Tagging and Dependency Parsing. In Trankit, given the detected sentences and tokens/words, we use a single model to jointly perform POS tagging, morphological feature tagging and dependency parsing at sentence level. Joint modeling mitigates error propagation, saves the memory, and speedups the system. In particular, given a sentence, the representation for each word is computed as the average of its wordpieces' transformer-based representations in $x_{1:K}^{l,m}$. Let $t_{1:N} = [t_1, t_2, \ldots, t_N]$ be the representations of the words in the sentence. We compute the following vectors using feed-forward networks FFN$_*$:

$$r_{1:N}^{upos} = \text{FFN}_{upos}(t_{1:N}), r_{1:N}^{xpos} = \text{FFN}_{xpos}(t_{1:N})$$

$$r_{1:N}^{ufeats} = \text{FFN}_{ufeats}(t_{1:N}), r_{0:N}^{dep} = [x_{cls}; \text{FFN}_{dep}(t_{1:N})]$$

Vectors for the words in $r_{1:N}^{upos}$, $r_{1:N}^{xpos}$, $r_{1:N}^{ufeats}$ are then passed to a softmax layer to make predictions for UPOS, XPOS, and UFeats tags for each word. For dependency parsing, we use the classification token `<s>` to represent the root node, and apply Deep Biaffine Attention (Dozat & Manning, 2017) and the Chu-Liu/Edmonds algorithm (Chu, 1965; Edmonds, 1967) to assign a syntactic head and the associated dependency relation to each word in the sentence.

Lemmatizer. This component receives sentences and their predicted UPOS tags to produce the canonical form for each word. We also employ a character-based seq2seq model for this component as in Stanza.

Named Entity Recognizer. Given a sentence, the named entity recognizer determines spans of entity names by assigning a BIOES tag to each token in the sentence. We deploy a standard sequence labeling architecture using transformer-based representations for tokens, involving a feed-forward network followed by a Conditional Random Field.

2.4 Usage

Detailed documentation for Trankit can be found at: `https://trankit.readthedocs.io`.

Trankit Installation. Trankit is written in Python and available on PyPI: `https://pypi.org/project/trankit/`. Users can install our toolkit via pip using:

```
pip install trankit
```

Initialize a Pipeline. Lines 1-4 in Figure 4 shows how to initialize a pretrained pipeline for English; it is instructed to run on GPU and store downloaded pretrained models to the specified cache directory. Trankit will not download pretrained models if they already exist.

Multilingual Usage. Figure 4 shows how to initialize a multilingual pipeline and process inputs of different languages in Trankit:

```python
from trankit import Pipeline

# initialize a multilingual pipeline
p = Pipeline(lang='english', gpu=True, cache_dir='./cache')
langs = ['arabic', 'chinese', 'dutch']
for lang in langs:
        p.add(lang)

# tokenize English input
p.set_active('english')
en = p.tokenize('Rich was here before the scheduled time.')

# get ner tags for Arabic input
p.set_active('arabic')
ar = p.ner('وكان كنعان قبل ذلك رئيس جهاز الامن والاستطلاع للقوات السورية العاملة في لبنان.')
```

Figure 4. Multilingual pipeline initialization.

Basic Functions. Trankit can process inputs which are untokenized (raw) or pretokenized strings, at both sentence and document levels. Figure 5 illustrates a simple code to perform all the supported tasks for an input text. We organize Trankit's outputs into hierarchical native Python dictionaries, which can be easily inspected by users. Figure 6 demonstrates the outputs of the command line 6 in Figure 5.

```python
1 from trankit import Pipeline
2
3 p = Pipeline(lang='english', gpu=True, cache_dir='./cache')
4
5 doc = '''Hello! This is Trankit.'''
6 # perform all tasks on the input
7 all = p(doc)
```

Figure 5. A function performing all tasks on the input.

Training your own Pipelines. Trankit also provides a trainable pipeline for 100 languages via the class `TPipeline`. This ability is inherited from the XLM-Roberta encoder which is pretrained on those languages. Figure 7 illustrates how to train a token and sentence splitter with `TPipeline`.

```
// Output
{
  'text': 'Hello! This is Trankit.', // input string
  'sentences': [ // list of sentences
    {
      'id': 1, 'text': 'Hello!', 'dspan': (0, 6), 'tokens': [...]
    },
    {
      'id': 2, // sentence index
      'text': 'This is Trankit.', 'dspan': (7, 23), // sentence span
      'tokens’: [ // list of tokens
        {
          'id': 1, // token index
          'text': 'This', 'upos': 'PRON', 'xpos': 'DT',
          'feats': 'Number=Sing|PronType=Dem',
          'head': 3, 'deprel': 'nsubj', 'lemma': 'this', 'ner': 'O',
          'dspan': (7, 11), // document-level span of the token
          'span': (0, 4)    // sentence-level span of the token
        },
        {'id': 2...},
        {'id': 3...},
        {'id': 4...}
      ]
    }
  ]
}
```

Figure 6. Output from Trankit. Some parts are collapsed to improve visualization.

```python
from trankit import TPipeline

tp = TPipeline(training_config={
         'task': 'tokenize',
         'save_dir': './saved_model',
         'train_txt_fpath': './train.txt',
         'train_conllu_fpath': './train.conllu',
         'dev_txt_fpath': './dev.txt',
         'dev_conllu_fpath': './dev.conllu'})

trainer.train()
```

Figure 7. Training a token and sentence splitter using the CONLL-U formatted data (Nivre et al., 2020).

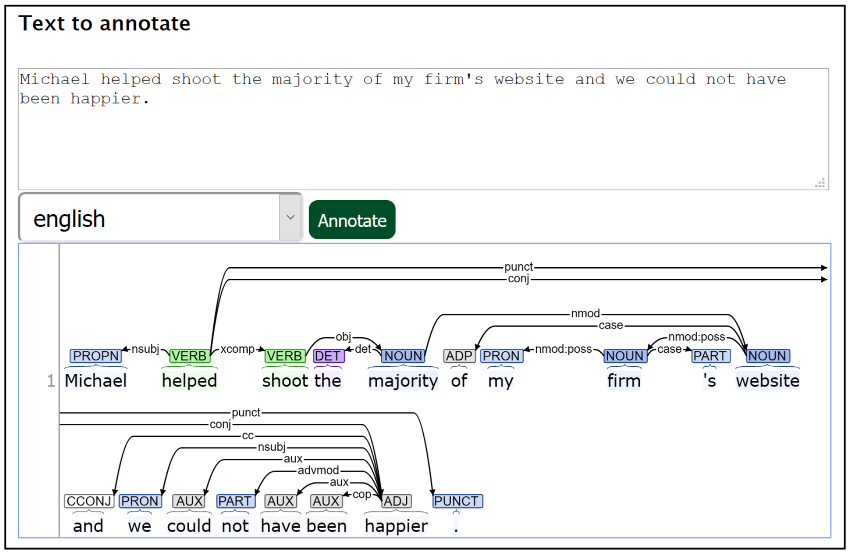
Demo Website. A demo website for Trankit to support 90 pretrained pipelines is hosted at: `http://nlp.uoregon.edu/trankit`. Figure 8 shows its interface.

2.5 System Evaluation

2.5.1 Datasets & Hyper-parameters.

To achieve a fair comparison, we follow Stanza (Qi et al., 2020b) to train and evaluate all the models on the same canonical data splits of 90 Universal Dependencies treebanks v2.5 (UD2.5)[3] (Zeman et al., 2019), and 11 public NER datasets provided in the following corpora: AQMAR (Mohit, Schneider, Bhowmick, Oflazer, & Smith, 2012), CoNLL02 (Tjong Kim Sang, 2002), CoNLL03 (Tjong Kim Sang & De Meulder, 2003), GermEval14

[3]We skip 10 treebanks whose languages are not supported by *XLM-Roberta*.


Text to annotate
english
Annotate
PROPN nsubj VERB xcomp VERB DET obj det NOUN ADP PRON nmod:poss NOUN case PART nmod:poss NOUN nmod case punct conj
Michael helped shoot the majority of my firm 's website
CCONJ PRON AUX PART AUX AUX ADJ PUNCT punct conj cc nsubj aux advmod aux cop
and we could not have been happier .

Michael helped shoot the majority of my firm's website and we could not have been happier.

Figure 8. Demo website for Trankit.

(Benikova, Biemann, & Reznicek, 2014), OntoNotes (Weischedel et al., 2013), and WikiNER (Nothman, Ringland, Radford, Murphy, & Curran, 2012). Hyper-parameters for all models and datasets are selected based on the development data in this work.

System	Tokens	Sents.	Words	UPOS	XPOS	UFeats	Lemmas	UAS	LAS
Trankit (with adapters)	**99.05**	**95.12**	**98.96**	**95.43**	**89.02**	**92.69**	**93.46**	**86.20**	**82.51**
Multilingual	96.69	88.95	96.35	91.19	84.64	88.10	90.02	72.96	68.66
No-adapters	95.06	89.57	94.08	88.79	82.54	83.76	88.33	66.63	63.11

Table 2. Model performance on 9 different treebanks (macro-averaged F1 score over test sets).

2.5.2 Universal Dependencies performance.

Table 1 compares the performance of Trankit and the latest available versions of other popular toolkits, including Stanza (v1.1.1) with current state-of-the-art performance, UDPipe (v1.2), and spaCy (v2.3) on the UD2.5 test sets. The performance for all systems is obtained using the official scorer of the CoNLL 2018 Shared Task[4]. On five illustrated languages, Trankit achieves competitive performance on tokenization,

[4]`https://universaldependencies.org/conll18/evaluation.html`

MWT expansion, and lemmatization. Importantly, Trankit outperforms other toolkits over all remaining tasks (e.g., POS and morphological tagging) in which the improvement boost is substantial and significant for sentence segmentation and dependency parsing. For example, English enjoys a 7.22% improvement for sentence segmentation, a 3.92% and 4.37% improvement for UAS and LAS in dependency parsing. For Arabic, Trankit has a remarkable improvement of 16.16% for sentence segmentation while Chinese observes 12.31% and 12.72% improvement of UAS and LAS for dependency parsing.

Over all 90 treebanks, Trankit outperforms the previous state-of-the-art framework Stanza in most of the tasks, particularly for sentence segmentation (+3.24%), POS tagging (+1.44% for UPOS and +1.55% for XPOS), morphological tagging (+1.46%), and dependency parsing (+4.0% for UAS and +5.01% for LAS) while maintaining the competitive performance on tokenization, multi-word expansion, and lemmatization.

2.5.3 NER results. Table 3 compares Trankit with Stanza (v1.1.1), Flair (v0.7), and spaCy (v2.3) on the test sets of 11 considered NER datasets. Following Stanza, we report the performance for other toolkits with their pretrained models on the canonical data splits if they are available. Otherwise, their best configurations are used to train the models on the same data splits (inherited from Stanza). Also, for the Dutch datasets, we retrain the models in Flair as those models (for Dutch) have been updated in version v0.7. As can be seen, Trankit obtains competitive or better performance for most of the languages, clearly demonstrating the benefit of using the pretrained transformer for multilingual NER.

Language	Corpus	**Trankit**	Stanza	Flair	spaCy
Arabic	AQMAR	**74.8**	74.3	74.0	-
Chinese	OntoNotes	**80.0**	79.2	-	69.3
Dutch	CoNLL02	**91.8**	89.2	91.3	73.8
	WikiNER	**94.8**	**94.8**	**94.8**	90.9
English	CoNLL03	92.1	92.1	**92.7**	81.0
	OntoNotes	**89.6**	88.8	89.0	85.4
French	WikiNER	92.3	**92.9**	92.5	88.8
German	CoNLL03	**84.6**	81.9	82.5	63.9
	GermEval14	**86.9**	85.2	85.4	68.4
Russian	WikiNER	92.8	**92.9**	-	-
Spanish	CoNLL02	**88.9**	88.1	87.3	77.5

Table 3. Performance (F1) on NER test sets.

System	GPU		CPU	
	UD	NER	UD	NER
Trankit	4.50×	1.36×	19.8×	31.5×
Stanza	3.22×	1.08×	10.3×	17.7×
UDPipe	-	-	4.30×	-
Flair	-	1.17×	-	51.8×

Table 4. Run time on processing the English EWT treebank and the English Ontonotes NER dataset. Measurements are done on an NVIDIA Titan RTX card.

2.5.4 Speed and Memory Usage. Table 4 reports the relative processing time for UD and NER of the toolkits compared to spaCy's CPU processing time[5]. For memory usage comparison, we show the model sizes of Trankit and Stanza for several languages in Table 5. As can be seen, besides the multilingual transformer, model packages in Trankit only take dozens of megabytes while Stanza consumes hundreds of megabytes for each package. This leads to the Stanza's usage of much more memory when the pipelines for these languages are loaded at the same time. In fact, Trankit only takes 4.9GB to load all the 90 pretrained pipelines for the 56 supported languages.

2.5.5 Ablation Study. This section compares Trankit with two other possible strategies to build a multilingual system for fundamental NLP tasks. In

[5]spaCy can process 8140 tokens and 5912 tokens per second for UD and NER, respectively.

Model Package	Trankit	Stanza
Multilingual Transformer	1146.9MB	-
Arabic	38.6MB	393.9MB
Chinese	40.6MB	225.2MB
English	47.9MB	383.5MB
French	39.6MB	561.9MB
Spanish	37.3MB	556.1MB
Total size	1350.9MB	2120.6MB

Table 5. Model sizes for five languages.

the first strategy (called "*Multilingual*"), we train a single pipeline where all the components in the pipeline are trained with the combined training data of all the languages. The second strategy (called "*No-adapters*") involves eliminating adapters from *XLM-Roberta* in Trankit. As such, in "*No-adapters*", pipelines are still trained separately for each language; the pretrained transformer is fixed; and only task-specific weights (for predictions) in components are updated during training.

For evaluation, we select 9 treebanks for 3 different groups, i.e., high-resource, medium-resource, and low-resource, depending on the sizes of the treebanks. In particular, the high-resource group includes Czech, Russian, and Arabic; the medium-resource group includes French, English, and Chinese; and the low-resource group involves Belarusian, Telugu, and Lithuanian. Table 2 compares the average performance of Trankit, "*Multilingual*", and "*No-adapters*". As can be seen, "*Multilingual*" and "*No-adapters*" are significantly worse than the proposed adapter-based Trankit. We attribute this to the fact that multilingual training might suffer from unbalanced sizes of treebanks, causing high-resource languages to dominate others and impairing the overall performance. For "*No-adapters*", fixing pretrained transformer might significantly limit the models' capacity for multiple tasks and languages.

2.6 Summary

We introduce Trankit, a transformer-based multilingual toolkit that significantly improves the performance for fundamental NLP tasks, including sentence segmentation, part-of-speech, morphological tagging, and dependency parsing over 90 Universal Dependencies v2.5 treebanks of 56 different languages. Our toolkit is fast on GPUs and efficient in memory use, making it usable for general users.

CHAPTER III

LANGUAGE-AGNOSTIC MODELS FOR JOINT INFORMATION EXTRACTION

This chapter contains materials from the published papers: *"Minh Nguyen, Viet Dac Lai, and Thien Huu Nguyen.* ***'Cross-Task Instance Representation Interactions and Label Dependencies for Joint Information Extraction with Graph Convolutional Networks'*** *In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2021"* (M. V. Nguyen, Lai, & Nguyen, 2021); *"Minh Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen.* ***'Learning Cross-Task Dependencies for Joint Extraction of Entities, Events, Event Arguments, and Relations'*** *In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 2022"* (M. V. Nguyen, Min, et al., 2022b); and *"Minh Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen.* ***'Joint Extraction of Entities, Relations, and Events via Modeling Inter-Instance and Inter-Label Dependencies'*** *In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2022"* (M. V. Nguyen, Min, et al., 2022a). Minh was responsible for the model design, experiments, evaluation and writing as the first author. Thien, Viet, Bonan, and Franck provided meaningful discussion and analysis. Thien contributed to the model design and editorial writing for the paper submissions. The papers were revised to comply with the dissertation format and purposes.

After introducing Trankit to enhance the upstream models for multilingual IE in chapter II, this chapter shifts the focus from linguistic feature processing to the architecture of IE models themselves (RD2). RD2 aims to develop models that can be universally applied across languages without requiring language-specific modifications. In this chapter, we introduce FourIE, DepIE, and GraphIE, our novel language-agnostic models developed to tackle the four tasks of IE within a unified framework. These models offer innovative contributions designed to capture the interdependencies between tasks effectively, improving upon previous efforts in joint IE. FourIE introduces an interaction graph and a dependency graph to capture cross-task dependencies at both the representation and label levels. DepIE improves upon FourIE by learning cross-task dependencies from data instead of manually defining them based on heuristics. GraphIE addresses limitations in prior joint IE models to better capture dependencies between task instances and their labels, utilizing learned dependency graphs, Conditional Random Fields, and Simulated Annealing for optimal performance. The models achieve state-of-the-art performance for joint IE on both monolingual and multilingual learning settings across various datasets and languages.

3.1 FourIE

3.1.1 Introduction.

Information Extraction (IE) is an important and challenging task in Natural Language Processing (NLP) that aims to extract structured information from unstructured texts. Following the terminology for IE in the popular ACE 2005 program (Walker, Strassel, Medero, & Maeda, 2006), we focus on four major IE tasks in this work: entity mention extraction (EME), relation extraction (RE), event trigger detection (ETD), and event argument extraction (EAE).

Given an input sentence, a vast majority of prior work has solved the four tasks in IE independently at both instance and task levels (called independent

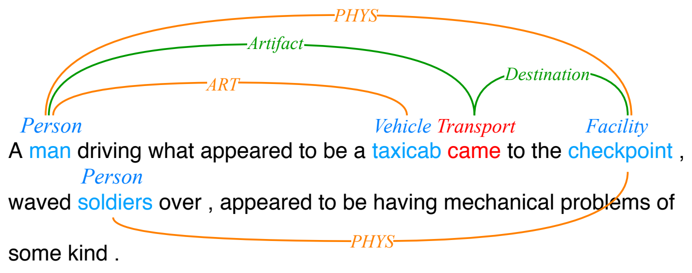
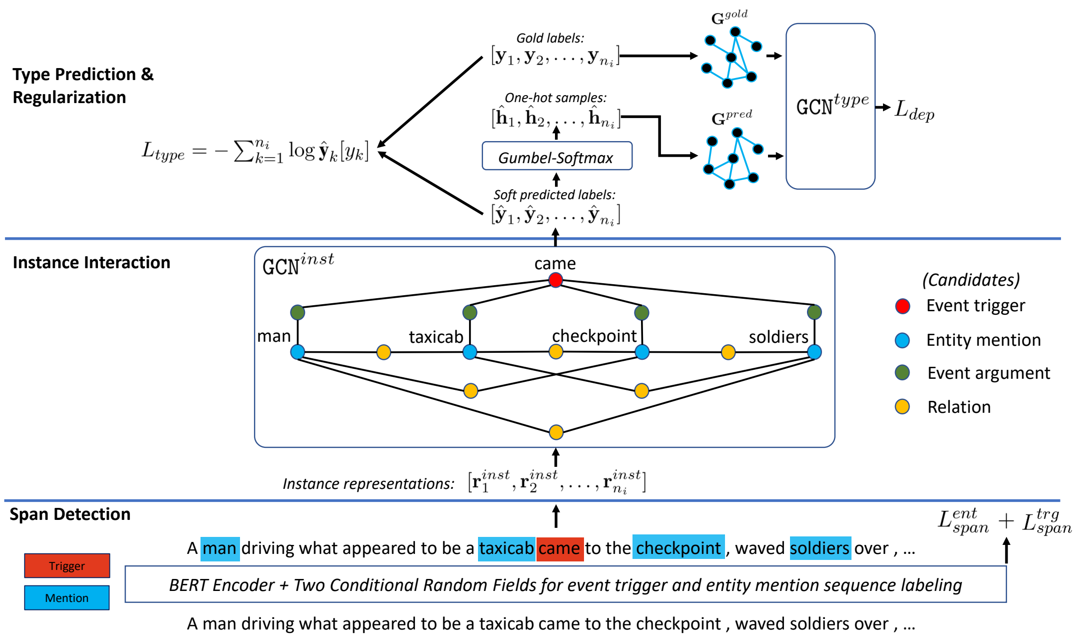

PHYS
Artifact
ART
Destination
Person
Vehicle
Transport
Facility
A man driving what appeared to be a taxicab came to the checkpoint ,
Person
waved soldiers over , appeared to be having mechanical problems of
PHYS
some kind .

Figure 9. A sentence example with the annotations for the four IE tasks. Blue words corresponds to entity mentions while red words are event triggers. Also, orange edges represent relations while green edges indicate argument roles.

prediction models). First, at the instance level, each IE task often requires predictions/classifications for multiple instances in a single input sentence. For instance, in RE, one often needs to predict relations for every pair of entity mentions (called relation instances) in the sentence while multiple word spans in the sentence can be viewed as multiple instances where event type predictions have to be made in ETD (trigger instances). As such, most prior work on IE has performed predictions for instances in a sentence separately by treating each instance as one example in the dataset (Y. Chen, Xu, Liu, Zeng, & Zhao, 2015a; V. D. Lai, Nguyen, & Nguyen, 2020; T. H. Nguyen & Grishman, 2015a, 2015c; Santos & Guimaraes, 2015; G. Zhou, Su, Zhang, & Zhang, 2005a). Second, at the task level, prior work on IE tends to perform the four tasks in a pipelined approach where outputs from one task are used as inputs for other tasks (e.g., EAE is followed by EME and ETD) (Y. Chen et al., 2015a; Q. Li, Ji, & Huang, 2013a; Veyseh, Nguyen, & Nguyen, 2020a).

Despite its popularity, the main issue of the independent prediction models is that they suffer from the error propagation between tasks and the failure to exploit the cross-task and cross-instance inter-dependencies within an input sentence to improve the performance for IE tasks. For instance, such systems are



Type Prediction & Regularization
$L_{type} = -\sum_{k=1}^{n_i} \log \hat{\mathbf{y}}_k[y_k]$
Gold labels:
$[\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_i}]$
One-hot samples:
$[\hat{\mathbf{h}}_1, \hat{\mathbf{h}}_2, \ldots, \hat{\mathbf{h}}_{n_i}]$
Gumbel-Softmax
Soft predicted labels:
$[\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, \ldots, \hat{\mathbf{y}}_{n_i}]$
$\mathbf{G}^{gold}$
$\mathbf{G}^{pred}$
$\mathtt{GCN}^{type}$
L_{dep}
Instance Interaction
$\mathtt{GCN}^{inst}$
came
man
taxicab
checkpoint
soldiers
(Candidates)
Event trigger
Entity mention
Event argument
Relation
Instance representations: $[\mathbf{r}_1^{inst}, \mathbf{r}_2^{inst}, \ldots, \mathbf{r}_{n_i}^{inst}]$
Span Detection
$L_{span}^{ent} + L_{span}^{trg}$
A man driving what appeared to be a taxicab came to the checkpoint , waved soldiers over , …
Trigger
Mention
BERT Encoder + Two Conditional Random Fields for event trigger and entity mention sequence labeling
A man driving what appeared to be a taxicab came to the checkpoint , waved soldiers over , …

Figure 10. Overall architecture of our proposed model. At the representation level, $\mathtt{GCN}^{inst}$ is used to enrich the representations for instances of the four tasks. At the label level, $\mathtt{GCN}^{type}$ is responsible for capturing the connections between the types in the dependency graphs, thus helping the model learn the structural difference between the gold graph $\mathbf{G}^{gold}$ and the predicted graph $\mathbf{G}^{pred}$.

unable to benefit from the dependency that the *Victim* of a *Die* event has a high chance to also be the *Victim* of an *Attack* event in the same sentence (i.e., type or label dependencies). To address these issues, some prior work has explored joint inference models where multiple tasks of IE are performed simultaneously for all task instances in a sentence, using both feature-based models (Q. Li et al., 2013a; Miwa & Sasaki, 2014; Roth & Yih, 2004a; B. Yang & Mitchell, 2016a) and recent deep learning models (Miwa & Bansal, 2016; Zhang, Qin, Zhang, Liu, & Ji, 2019). However, such prior work has mostly considered joint models for a subset of the four IE tasks (e.g., EME+RE or ETD+EAE), thus still suffering from the error propagation issue (with the missing tasks) and failing to fully exploit potential inter-dependencies between the four tasks. To this end, this work aims to design a single model to simultaneously solve the four IE tasks for each input sentence (joint four-task IE) to address the aforementioned issues of prior joint IE work.

Few recent work has considered joint four-task IE, using deep learning to produce state-of-the-art (SOTA) performance for the tasks (Y. Lin, Ji, Huang, & Wu, 2020a; Wadden, Wennberg, Luan, & Hajishirzi, 2019a). However, there are still two problems that hinder further improvement of such models. First, at the instance level, an important component of deep learning models for joint IE involves the representation vectors of the instances that are used to perform the corresponding prediction tasks for IE in an input sentence (called predictive instance representations). For joint four-task IE, we argue that there are inter-dependencies between predictive representation vectors of related instances for the four tasks that should be modeled to improve the performance for IE. For instance, the entity type information encoded in the predictive representation vector for an entity mention can constrain the argument role that the representation vector for a related EAE instance (e.g., involving the same entity mention and some event trigger in the same sentence) should capture and vice versa. As such, prior work for joint four-task IE has only computed predictive representation vectors for instances of the tasks independently using shared hidden vectors from some deep learning layer (Y. Lin et al., 2020a; Wadden et al., 2019a). Although this shared mechanism helps capture the interaction of predictive representation vectors to some extent, it fails to explicitly present the connections between related instances of different tasks and encode them into the representation learning process. Consequently, to overcome this issue, we propose a novel deep learning model for joint four-task IE (called *FourIE*) that creates a graph structure to explicitly capture the interactions between related instances of the four IE tasks in a sentence. This graph will then be consumed by a graph convolutional network (GCN) (Kipf & Welling, 2017; T. H. Nguyen & Grishman, 2018a) to enrich the representation vector for an instance with those from the related (neighboring) instances for IE.

Second, at the task level, existing joint four-task models for IE have only exploited the cross-task type dependencies in the decoding step to constrain predictions for the input sentence (by manually converting the type dependency graphs of the input sentence into global feature vectors for scoring the predictions in the beam search-based decoding) (Y. Lin et al., 2020a). The knowledge from cross-task type dependencies thus cannot contribute to the training process of the IE models. This is unfortunate as we expect that deeper integration of this knowledge into the training process could provide useful information to enhance representation learning for IE tasks. To this end, we propose to use the knowledge from cross-task type dependencies to obtain an additional training signal for each sentence to directly supervise our joint four-task IE model. In particular, our motivation is that the types expressed in a sentence for the four IE tasks can be organized into a dependency graph between the types (global type dependencies for the sentence). As such, in order for a joint model to perform well, the type dependency graph generated by its predictions for a sentence should be similar to the dependency graph obtained from the golden types (i.e., a global type constraint on the predictions in the training step). A novel regularization term is thus introduced into the training loss of our joint model to encode this constraint, employing another GCN to learn representation vectors for the predicted and golden dependency graphs to facilitate the graph similarity promotion. To our knowledge, this is the first work that employs global type dependencies to regularize joint models for IE.

Finally, our extensive experiments demonstrate the effectiveness of the proposed model on benchmark datasets in three different languages (e.g., English, Chinese, and Spanish), leading to state-of-the-art performance on different settings.

3.1.2 Problem Statement and Background. The joint four-task IE problem in this work takes a sentence as the input and aims to jointly solve four tasks EAE, ETD, RE, and EAE using an unified model. As such, the goal of EME is to detect and classify entity mentions (names, nominals, pronouns) according to a set of predefined (semantic) entity types (e.g., *Person*). Similarly, ETD seeks to identify and classify event triggers (verbs or normalization) that clearly evoke an event in some predefined set of event types (e.g., *Attack*). Note that event triggers can involve multiple words. For RE, its concern is to predict the semantic relationship between two entity mentions in the sentence. Here, the set of relations of interest is also predefined and includes a special type of *None* to indicate *no-relation*. Finally, in EAE, given an event trigger, the systems need to predict the roles (also in a predefined set with a special type *None*) that each entity mention plays in the corresponding event. Entity mentions are thus also called event argument candidates in this work. Figure 9 presents a sentence example where the expected outputs for each IE task are illustrated.

Graph Convolutional Networks (GCN): As GCNs are used extensively in our model, we present their computation process in this section to facilitate the discussion. Given a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ where $\mathbf{V} = \{v_1, \ldots, v_u\}$ is the node set (with u nodes) and $\mathbf{E}$ is the edge set. In GCN, the edges in $\mathbf{G}$ are often captured via the adjacency matrix $\mathbf{A} \in \mathbb{R}^{u \times u}$. Also, each node $v_i \in \mathbf{V}$ is associated with an initial hidden vector $\mathbf{v}_i^0$. As such, a GCN model involves multiple layers of abstraction in which the hidden vector $\mathbf{v}_i^l$ for the node $v_i \in \mathbf{V}$ at the l-th layer is computed by ($l \geq 1$):

$$\mathbf{v}_i^l = \text{ReLU}\left(\frac{\sum_{j=1}^{u} \mathbf{A}_{ij} \mathbf{W}^l \mathbf{v}_j^{l-1} + \mathbf{b}^l}{\sum_{j=1}^{u} \mathbf{A}_{ij}}\right)$$

where $\mathbf{W}^l$ and $\mathbf{b}^l$ are trainable weight and bias at the l-th layer. Assuming N GCN layers, the hidden vectors for the nodes in $\mathbf{V}$ at the last layer $\mathbf{v}_1^N, \ldots, \mathbf{v}_u^N$ would capture richer and more abstract information for the nodes, serving as the outputs of the GCN model. This process is denoted by: $\mathbf{v}_1^N, \ldots, \mathbf{v}_u^N = \texttt{GCN}(\mathbf{A}; \mathbf{v}_1^0, \ldots, \mathbf{v}_u^0; N)$.

3.1.3 Model. Given an input sentence $\mathbf{w} = [w_1, w_2, \ldots, w_n]$ (with n words), our model for joint four-task IE on $\mathbf{w}$ involves three major components: (i) Span Detection, (ii) Instance Interaction, and (iii) Type Dependency-based Regularization.

Span Detection: This component aims to identify spans of entity mentions and event triggers in $\mathbf{w}$ that would be used to form the nodes in the interaction graph between different instances of our four IE tasks for $\mathbf{w}$. As such, we formulate the span detection problems as sequence labeling tasks where each word w_i in $\mathbf{w}$ is associated with two BIO tags to capture the span information for entity mentions and event triggers in $\mathbf{w}$. Note that we do not predict entity types and event types at this step, leading to only three possible values (i.e., B, I, and O) for the tags of the words.

In particular, following (Y. Lin et al., 2020a), we first feed $\mathbf{w}$ into the pre-trained BERT encoder (Devlin, Chang, Lee, & Toutanova, 2019a) to obtain a sequence of vectors $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ to represent $\mathbf{w}$. Here, each vector $\mathbf{x}_i$ serves as the representation vector for the word $w_i \in \mathbf{w}$ that is obtained by averaging the hidden vectors of the word-pieces of w_i returned by BERT. Afterward, $\mathbf{X}$ is fed into two conditional random field (CRF) layers to determine the best BIO tag sequences for event mentions and event triggers for $\mathbf{w}$, following (Chiu & Nichols, 2016). As such, the Viterbi algorithm is used to decode the input sentence while the negative log-likelihood losses are employed as the training objectives for the span detection component of the model. For convenience, let L^{ent}_{span} and L^{trg}_{span} be the negative log-likelihoods of the gold tag sequences for entity mentions and event triggers (respectively) for $\mathbf{w}$. These terms will be included in the overall loss function of the model later.

Instance Interaction: Based on the tag sequences for $\mathbf{w}$ from the previous component, we can obtain two separate span sets for the entity mentions and event triggers in $\mathbf{w}$ (the golden spans are used in the training phase to avoid noise). For the next computation, we first compute a representation vector for each span (i, j) $(1 \leq i \leq j \leq n)$ in these two sets by averaging the BERT-based representation vectors for the words in this span (i.e., $\mathbf{x}_i, \ldots, \mathbf{x}_j$). For convenience, let $\mathbf{R}^{ent} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{n_{ent}}\}$ $(n_{ent} = |\mathbf{R}^{ent}|)$ and $\mathbf{R}^{trg} = \{\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_{n_{trg}}\}$

($n_{trg} = |\mathbf{R}^{trg}|$) be the sets of span representation vectors for the entity mentions and event triggers in $\mathbf{w}$[1]. The goal of this component is to leverage such span representation vectors to form instance representations and enrich them with instance interactions to perform necessary predictions in IE.

Instance Representation: Prediction instances in our model amount to the specific objects that we need to predict a type for one of the four IE tasks. As such, the prediction instances for EME and ETD, called entity and trigger instances, correspond directly to the entity mentions and event triggers in $\mathbf{R}^{ent}$ and $\mathbf{R}^{trg}$ respectively (as we need to predict the entity types for $\mathbf{e}_i \in \mathbf{R}^{ent}$ and the event types for $\mathbf{t}_i \in \mathbf{R}^{trg}$ in this step). Thus, we also use $\mathbf{R}^{ent}$ and $\mathbf{R}^{trg}$ as the sets of initial representation vectors for the entity/event instances for EME and ETD in the following. Next, for RE, the prediction instances (called relation instances) involve pairs of entity mentions in $\mathbf{R}^{ent}$. To obtain the initial representation vector for a relation instance, we concatenate the representation vectors of the two corresponding entity mentions, leading to the set of representation vectors $\mathbf{rel}_{ij}$ for relation instances: $\mathbf{R}^{rel} = \{\mathbf{rel}_{ij} = [\mathbf{e}_i, \mathbf{e}_j] \mid \mathbf{e}_i, \mathbf{e}_j \in \mathbf{R}^{ent}, i < j\}$ ($|\mathbf{R}^{rel}| = n_{ent}(n_{ent} - 1)/2$). Finally, for EAE, we form the prediction instances (called argument instances) by pairing each event trigger in $\mathbf{R}^{trg}$ with each entity mention in $\mathbf{R}^{ent}$ (for the argument role predictions of the entity mentions with respect to the event triggers/mentions). By concatenating the representation vectors of the paired entity mentions and event triggers, we generate the initial representation vectors $\mathbf{arg}_{ij}$ for the corresponding argument instances: $\mathbf{R}^{arg} = \{\mathbf{arg}_{ij} = [\mathbf{t}_i, \mathbf{e}_j] \mid \mathbf{t}_i \in$

[1]We will also refer to entity mentions and event triggers interchangeably with their span representations $\mathbf{e}_i$ and $\mathbf{t}_i$ in this work.

$\mathbf{R}^{trg}, \mathbf{e}_j \in \mathbf{R}^{ent}\}$ ($|\mathbf{R}^{arg}| = n_{trg}n_{ent}$)[2]. We also use the prediction instances and their representation vectors interchangeably in this work.

Instance Interaction: The initial representation vectors for the instances so far do not explicitly consider beneficial interactions between related instances. To address this issue, we explicitly create an interaction graph between the prediction instances for the four IE tasks to connect related instances to each other. This graph will be consumed by a GCN model to enrich instance representations with interaction information afterward. In particular, the node set $\mathbf{N}^{inst}$ in our instance interaction graph $\mathbf{G}^{inst} = \{\mathbf{N}^{inst}, \mathbf{E}^{inst}\}$ involves all prediction instances for the four IE tasks, i.e., $\mathbf{N}^{inst} = \mathbf{R}^{ent} \cup \mathbf{R}^{trg} \cup \mathbf{R}^{rel} \cup \mathbf{R}^{arg}$. The edge set $\mathbf{E}^{inst}$ then captures instance interactions by connecting the instance nodes in $\mathbf{N}^{inst}$ that involve the same entity mentions or event triggers (i.e., two instances are related if they concern the same entity mention or event trigger). As such, the edges in $\mathbf{E}^{inst}$ are created as follows:

(i) An entity instance node $\mathbf{e}_i$ is connected to all relation instance nodes of the forms $\mathbf{rel}_{ij} = [\mathbf{e}_i, \mathbf{e}_j]$ and $\mathbf{rel}_{ki} = [\mathbf{e}_k, \mathbf{e}_i]$ (sharing entity mention $\mathbf{e}_i$).

(ii) An entity instance node $\mathbf{e}_j$ is connected to all argument instance nodes of the form $\mathbf{arg}_{ij} = [\mathbf{t}_i, \mathbf{e}_j]$ (sharing entity mention $\mathbf{e}_j$).

(iii) A trigger node $\mathbf{t}_i$ is connected to all argument instance nodes of the form $\mathbf{arg}_{ij} = [\mathbf{t}_i, \mathbf{e}_j]$ (i.e., sharing event trigger $\mathbf{t}_i$).

GCN: To enrich the representation vector for an instance in $\mathbf{N}^{inst}$ with the information from the related (neighboring) nodes, we feed $\mathbf{G}^{inst}$ into a GCN model (called $\mathtt{GCN}^{inst}$). For convenience, we rename the initial representation vectors of all the instance nodes in $\mathbf{N}^{inst}$ by: $\mathbf{N}^{inst} = \{\mathbf{r}_1, \ldots, \mathbf{r}_{n_i}\}$ ($n_i = |\mathbf{N}^{inst}|$). Also, let $\mathbf{A}^{inst} \in$

[2]In our implementation, $\mathbf{R}^{rel}$ and $\mathbf{R}^{arg}$ are transformed into vectors of the same size with those in $\mathbf{R}^{ent}$ and $\mathbf{R}^{trg}$ (using one-layer feed forward networks) for future computation.

$\{0,1\}^{n_i \times n_i}$ be the adjacency matrix of the interaction graph $\mathbf{G}^{inst}$ where $\mathbf{A}^{inst}_{ij} = 1$ if the instance nodes $\mathbf{r}_i$ and $\mathbf{r}_j$ are connected in $\mathbf{G}^{inst}$ or $i = j$ (for self-connections). The interaction-enriched representation vectors for the instances in $\mathbf{N}^{inst}$ are then computed by the $\mathtt{GCN}^{inst}$ model: $\mathbf{r}^{inst}_1, \ldots, \mathbf{r}^{inst}_{n_i} = \mathtt{GCN}^{inst}(\mathbf{A}^{inst}; \mathbf{r}_1, \ldots, \mathbf{r}_{n_i}; N_i)$ where N_i is the number of layers for the $\mathtt{GCN}^{inst}$ model.

Type Embedding and Prediction: Finally, the enriched instance representation vectors $\mathbf{r}^{inst}_1, \ldots, \mathbf{r}^{inst}_{n_i}$ will be used to perform the predictions for the four IE tasks. In particular, let $t_k \in \{ent, trg, rel, arg\}$ be the corresponding task index and y_k be the ground-truth type (of the task t_k) for the prediction instance $\mathbf{r}_k$ in $\mathbf{N}^{inst}$. Also, let $\mathcal{T} = \mathcal{T}^{ent} \cup \mathcal{T}^{trg} \cup \mathcal{T}^{rel} \cup \mathcal{T}^{arg}$ be the union of the possible entity types (in $\mathcal{T}^{ent}$ for EME), event types (in $\mathcal{T}^{trg}$ for ETD), relations (in $\mathcal{T}^{rel}$ for RE), and argument roles (in $\mathcal{T}^{arg}$ for EAE) in our problem ($y_k \in \mathcal{T}^{t_k}$). Note that $\mathcal{T}^{rel}$ and $\mathcal{T}^{arg}$ contain the special types *None*. To prepare for the type predictions and the type dependency modeling in the next steps, we associate each type in $\mathcal{T}$ with an embedding vector (of the same size as $\mathbf{e}_i$ and $\mathbf{t}_i$) that is initialized randomly and updated during our training process. For convenience, let $\mathcal{T} = [\bar{\mathbf{t}}_1, \ldots, \bar{\mathbf{t}}_{n_t}]$ where $\bar{\mathbf{t}}_i$ is used interchangeably for both a type and its embedding vector in $\mathcal{T}$ (n_t is the total number of types). As such, to perform the prediction for an instance $\mathbf{r}_k$ in $\mathbf{N}^{inst}$, we compute the dot products between $\mathbf{r}^{inst}_k$ and each type embedding vectors in $\mathcal{T}^{t_k} \cap \mathcal{T}$ to estimate the possibilities that $\mathbf{r}_k$ has a type in $\mathcal{T}^{t_k}$. Afterward, these scores are normalized by the softmax function to obtain the probability distribution $\hat{\mathbf{y}}_k$ over the possible types in $\mathcal{T}^{t_k}$ for $\mathbf{r}_k$: $\hat{\mathbf{y}}_k = softmax(\mathbf{r}^{inst}_k \bar{\mathbf{t}}^T | \bar{\mathbf{t}} \in \mathcal{T}^{t_k} \cap \mathcal{T})$. In the decoding phase, the predicted type $\hat{y}_k$ for $\mathbf{r}_k$ is obtained via the $\mathtt{argmax}$ function (greedy decoding): $\hat{y}_k = \mathtt{argmax}\ \hat{\mathbf{y}}_k$. The negative log-likelihood over all the prediction instances is used to train the model: $L_{type} = -\sum_{k=1}^{n_i} \log \hat{\mathbf{y}}_k[y_k]$.

Type Dependency-based Regularization: In this section, we aim to obtain the type dependencies across tasks and use them to supervise the model in the training process (to improve the representation vectors for IE). As presented in the introduction, our motivation is to generate global dependency graphs between types of different IE tasks for each input sentence whose representations are leveraged to regularize the model during training. In particular, starting with the golden types $\mathbf{y} = y_1, y_2, \ldots, y_{n_i}$ and the predicted types $\hat{\mathbf{y}} = \hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{n_i}$ for the instance nodes in $\mathbf{N}^{inst}$, we build two dependency graphs $\mathbf{G}^{gold}$ and $\mathbf{G}^{pred}$ to capture the global type dependencies for the tasks (called the golden and predicted dependency graphs respectively). Afterward, to supervise the training process, we seek to constrain the model so the predicted dependency graph $\mathbf{G}^{pred}$ is similar to the golden graph $\mathbf{G}^{gold}$ (i.e., using the dependency graphs as the bridges to inject the global type dependency knowledge in $\mathbf{G}^{gold}$ into the model).

Dependency Graph Construction. Both $\mathbf{G}^{gold}$ and $\mathbf{G}^{pred}$ involve the types of all the four IE tasks in $\mathcal{T}$ as the nodes. To encode the type dependencies, the connections/edges in $\mathbf{G}^{gold}$ are computed based on the golden types $\mathbf{y} = y_1, y_2, \ldots, y_{n_i}$ for the instance nodes in $\mathbf{N}^{inst}$ as follows:

(i) For each relation instance node $\mathbf{r}_k = [\mathbf{e}_i, \mathbf{e}_j] \in \mathbf{N}^{inst}$ that has the golden type $y_k \neq None$, the relation type node y_k is connected to the nodes of the golden entity types for the corresponding entity mentions $\mathbf{e}_i$ and $\mathbf{e}_j$ (called **entity_relation type edges**).

(ii) For each argument instance node $\mathbf{r}_k = [\mathbf{t}_i, \mathbf{e}_j]$ that has the role type $y_k \neq None$, the role type node y_k is connected to both the node for the golden event type of $\mathbf{t}_i$ (called **event_argument type edges**) and the node for the golden entity type of $\mathbf{e}_j$ (called **entity_argument type edges**).

The same procedure can be applied to build the predicted dependency graph $\mathbf{G}^{pred}$ based on the predicted types $\hat{\mathbf{y}} = \hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{n_i}$. Also, for convenience, let $\mathbf{A}^{gold}$ and $\mathbf{A}^{pred}$ (of size $n_t \times n_t$) be the binary adjacency matrices of $\mathbf{G}^{gold}$ and $\mathbf{G}^{pred}$ (including the self-loops) respectively.

Regularization: In the next step, we obtain the representation vectors for the dependency graphs $\mathbf{G}^{gold}$ and $\mathbf{G}^{pred}$ by feeding them into a GCN model (called $\texttt{GCN}^{type}$). This GCN model has N_t layers and uses the initial type embeddings $\mathcal{T} = [\bar{\mathbf{t}}_1, \ldots, \bar{\mathbf{t}}_{n_t}]$ as the inputs. In particular, the outputs of $\texttt{GCN}^{type}$ for the two graphs involve $\bar{\mathbf{t}}_1^{gold}, \ldots, \bar{\mathbf{t}}_{n_t}^{gold} = \texttt{GCN}^{type}(\mathbf{A}^{gold}; \bar{\mathbf{t}}_1, \ldots, \bar{\mathbf{t}}_{n_t}; N_t)$ and $\bar{\mathbf{t}}_1^{pred}, \ldots, \bar{\mathbf{t}}_{n_t}^{pred} = \texttt{GCN}^{type}(\mathbf{A}^{pred}; \bar{\mathbf{t}}_1, \ldots, \bar{\mathbf{t}}_{n_t}; N_t)$ that encode the underlying information for the type dependencies presented in $\mathbf{G}^{gold}$ and $\mathbf{G}^{pred}$. Finally, to promote the similarity of the type dependencies in $\mathbf{G}^{gold}$ and $\mathbf{G}^{pred}$, we introduce the mean square difference between their $\texttt{GCN}^{type}$-induced representation vectors into the overall loss function for minimization: $L_{dep} = \sum_{i=1}^{n_t} ||\bar{\mathbf{t}}_i^{gold} - \bar{\mathbf{t}}_i^{pred}||_2^2$.

Our final training loss is thus: $L = L_{span}^{ent} + L_{span}^{trg} + L_{type} + \lambda L_{dep}$ (λ is a trade-off parameter).

Approximating $\mathbf{A}^{pred}$: We distinguish two types of parameters in our model so far, i.e., the parameters used to compute instance representations, e.g., those in BERT and $\mathbf{G}^{inst}$ (called θ^{inst}), and the parameters for type dependency regularization, i.e., those for the type embeddings $\bar{\mathbf{t}}_1, \ldots, \bar{\mathbf{t}}_{n_t}$ and $\mathbf{G}^{type}$ (called θ^{dep}). As such, the current implementation only enables the training signal from L_{dep} to back-propagate to the parameters θ^{dep} and disallows L_{dep} to influence the instance representation-related parameters θ^{inst}. To enrich the instance representation vectors with type dependency information, we expect L_{dep} to be deeper integrated into the model by also contributing to θ^{inst}. To achieve this goal, we note that the block of back-propagation between L_{dep} and θ^{inst} is due to their only connection in the model via the adjacency matrix $\mathbf{A}^{pred}$, whose values are either one or zero. As such, the values in $\mathbf{A}^{pred}$ are not directly dependent on any parameter in θ^{inst}, making it impossible for the back-propagation to flow. To this end, we propose to approximate $\mathbf{A}^{pred}$ with a new matrix $\hat{\mathbf{A}}^{pred}$ that directly involves θ^{inst} in its values. In particular, let $\mathbf{I}^{inst}$ be the index set of the non-zero cells in $\mathbf{A}^{pred}$: $\mathbf{I}^{inst} = \{(i,j)|\mathbf{A}^{pred}_{ij} = 1\}$. As the elements in $\mathbf{I}^{inst}$ are determined by the indexes $i_1, \ldots, i_{n_i}$ in $\mathcal{T}$ of the predicted types $\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{n_i}$ (respectively), we also seek to compute the values for the approximated matrix $\hat{\mathbf{A}}^{pred}$ based on such indexes. Accordingly, we first define the matrix $\mathbf{B} = \{b_{ij}\}_{i,j=1..n_t}$ where the element b_{ij} at the i-th row and j-th column is set to $b_{ij} = i * n_t + j$. The approximated matrix $\hat{\mathbf{A}}^{pred}$ is then obtained by:

$$\hat{\mathbf{A}}^{pred} = \sum_{(i,j)\in\mathbf{I}^{inst}} \exp\bigl(-\beta(\mathbf{B} - in_t - j)^2\bigr) \tag{3.1}$$

Here, $\beta > 0$ is a large constant. For each element $(i,j) \in \mathbf{I}^{inst}$, all the elements in the matrix $(\mathbf{B} - in_t - j)^2$ are strictly positive, except for the element at (i,j), which is zero. Thus, with a large value for β, the matrix $\exp(-\beta(\mathbf{B} - in_t - j)^2)$ has the value of one at cell (i,j) and nearly zero at other cells. Consequently, the values of $\hat{\mathbf{A}}^{pred}$ at the positions in $\mathbf{I}^{inst}$ are close to one while those at other positions are close to zero, thus approximating our expected matrix $\mathbf{A}^{pred}$ and still directly depending on the indexes $i_1, \ldots, i_{n_t}$.

Addressing the Discreteness of Indexes: Even with the approximation $\hat{\mathbf{A}}^{pred}$, the back-propagation still cannot flow from L_{dep} to θ^{inst} due to the block of the discrete and non-differentiable index variables $i_1, \ldots, i_{n_t}$. To address this issue, we propose to apply the Gumbel-Softmax distribution (Jang, Gu, & Poole, 2017) that enables the optimization of models with discrete random variables, by providing a method to approximate one-hot vectors sampled from a categorical distribution with continuous ones.

In particular, we first rewrite each index i_k by: $i_k = \mathbf{h}_k \mathbf{c}_k^T$, where $\mathbf{c}_k$ is a vector whose each dimension contains the index of a type in $\mathcal{T}^{t_k}$ in the joint type set $\mathcal{T}$, and $\mathbf{h}_k$ is the binary one-hot vector whose dimensions correspond to the types in $\mathcal{T}^{t_k}$. $\mathbf{h}_k$ is only turned on at the position corresponding to the predicted type $\hat{y}_k \in \mathcal{T}^{t_k}$ (indexed at i_k in $\mathcal{T}$). In our current implementation, $\hat{y}_k$ (thus the index i_k and the one-hot vector $\mathbf{h}_k$) is obtained via the `argmax` function: $\hat{y}_k = \texttt{argmax}\ \hat{\mathbf{y}}_k$, which causes the discreteness. As such, the Gumbel-Softmax distribution method helps to relax `argmax` by approximating $\mathbf{h}_k$ with a sample $\hat{\mathbf{h}}_k = \hat{h}_{k,1}, \ldots, \hat{h}_{k,|\mathcal{T}^{t_k}|}$ from the Gumbel-Softmax distribution:

$$\hat{h}_{k,j} = \frac{\exp((\log(\pi_{k,j}) + g_j)/\tau)}{\sum_{j'=1}^{|\mathcal{T}^{t_k}|} \exp((\log(\pi_{k,j'}) + g_{j'})/\tau)} \tag{3.2}$$

where $\pi_{k,j} = \hat{\mathbf{y}}_{k,j} = softmax_j(\mathbf{r}_k^{inst}\bar{\mathbf{t}}^T | \bar{\mathbf{t}} \in \mathcal{T}^{t_k} \cap \mathcal{T})$, $g_1, \ldots, g_{|\mathcal{T}^{t_k}|}$ are the i.i.d samples drawn from Gumbel(0,1) distribution (Gumbel, 1948): $g_j = -\log(-\log(u_j))$ ($u_j \sim \text{Uniform}(0,1)$), and τ is the temperature parameter. As $\tau \to 0$, the sample $\hat{\mathbf{h}}_k$ would become close to our expected one-hot vector $\mathbf{h}_k$. Finally, we replace $\mathbf{h}_k$ with the approximation $\hat{\mathbf{h}}_k$ in the computation for i_k: $i_k = \hat{\mathbf{h}}_k \mathbf{c}_k^T$ that directly depends on $\mathbf{r}_k^{inst}$ and is applied in $\hat{\mathbf{A}}^{pred}$. This allows the gradients to flow from L_{dep} to the parameters θ^{inst} and completes the description of our model.

3.1.4 Experiments. **Datasets**. Following the prior work on joint four-task IE (Y. Lin et al., 2020a; Wadden et al., 2019a), we evaluate our joint IE model (FourIE) on the ACE 2005 (Walker et al., 2006) and ERE datasets that provide annotation for entity mentions, event triggers, relations, and argument roles. In particular, we use three different versions of the ACE 2005 dataset that focus on three major joint inference settings for IE: (i) **ACE05-R** for joint inference of EME and RE, (ii) **ACE05-E** for joint inference of EME, ETD and EAE, and (iii) **ACE05-E+** for joint inference of the four tasks EME, ETD, RE, and EAE. ACE05-E+ is our main evaluation setting as it fits to our model design with the four IE tasks of interest.

Datasets	**Split**	**sents**	**ents**	**rels**	**events**
ACE05-R	Train	10,051	26,473	4,788	-
	Dev	2,424	6,362	1,131	-
	Test	2,050	5,476	1,151	-
ACE05-E	Train	17,172	29,006	4,664	4,202
	Dev	923	2,451	560	450
	Test	832	3,017	636	403
ACE05-E+	Train	19,240	47,525	7,152	4,419
	Dev	902	3,422	728	468
	Test	676	3,673	802	424
ERE-EN	Train	14,219	38,864	5,045	6,419
	Dev	1,162	3,320	424	552
	Test	1,129	3,291	477	559
ACE05-CN	Train	6,841	29,657	7,934	2,926
	Dev	526	2,250	596	217
	Test	547	2,388	672	190
ERE-ES	Train	7,067	11,839	1,698	3,272
	Dev	556	886	120	210
	Test	546	811	108	269

Table 6. Numbers of sentences (i.e., **sents**), entity mentions (i.e., **ents**), relations (i.e., **rels**), and events (i.e., **events**) in the datasets.

For ERE, following (Y. Lin et al., 2020a), we combine the data from three datasets for English (i.e., LDC2015E29, LDC2015E68, and LDC2015E78) that are created under the Deep Exploration and Filtering of Test (DEFT) program (called **ERE-EN**). Similar to ACE05-E+, ERE-EN is also used to evaluate the joint models on four IE tasks.

To demonstrate the portability of our model to other languages, we also apply FourIE to the joint four-IE datasets on Chinese and Spanish. Following (Y. Lin et al., 2020a), we use the ACE 2005 dataset for the evaluation on Chinese (called **ACE05-CN**) and the ERE dataset (LDC2015E107) for Spanish (called **ERE-ES**).

To ensure a fair comparison, we adopt the same data pre-processing and splits (train/dev/test) in prior work (Y. Lin et al., 2020a) for all the datasets. As such, ACE05-R, ACE05-E, ACE05-E+, and AC05-CN involve 7 entity types, 6 relation types, 33 event types, and 22 argument roles while ERE-ES and ERE-EN include 7 entity types, 5 relation types, 38 event types, and 20 argument roles. The statistics for the datasets are shown in Table 6.

Hyper-parameters and Evaluation Criteria. We fine-tune the hyper-parameters for our model using the development data. The suggested values are shown in the appendix. To achieve a fair comparison with (Y. Lin et al., 2020a), we employ the *bert-large-cased* model for the English datasets and *bert-multilingual-cased* model for the Chinese and Spanish datasets. Finally, we follow the same evaluation script and correctness criteria for entity mentions, event triggers, relations, and argument as in prior work (Y. Lin et al., 2020a). The reported results are the average performance of 5 model runs using different random seeds.

Performance Comparison. We compare the proposed model **FourIE** with two prior models for joint four-task IE: (i) **DyGIE++** (Wadden et al., 2019a): a BERT-based model with span graph propagation, and (ii) **OneIE** (Y. Lin et al., 2020a): *the current state-of-the-art (SOTA) model* for joint four-task IE based on BERT and type dependency constraint at the decoding step. Table 7 presents the performance (F1 scores) of the models on the test data of the English datasets. Note that in the tables, the prefixes "Ent", "Trg", "Rel", and "Arg" represent the extraction tasks for entity mentions, event triggers, relations, and arguments respectively while the suffixes "-I" and "-C" correspond to the identification performance (only concerning the offset correctness) and identification+classification performance (evaluating both offsets and types).

Datasets	**Task**	**DyGIE++**	**OneIE**	**FourIE**	Δ%
ACE05-R	Ent-C	88.6	88.8	**88.9**	0.1
	Rel-C	63.4	67.5	**68.9**†	1.4
ACE05-E	Ent-C	89.7	90.2	**91.3**†	1.1
	Trg-I	-	78.2	**78.3**	0.1
	Trg-C	69.7	74.7	**75.4**†	0.7
	Arg-I	53.0	59.2	**60.7**†	1.5
	Arg-C	48.8	56.8	**58.0**†	1.2
ACE05-E+	Ent-C	-	89.6	**91.1**†	1.5
	Rel-C	-	58.6	**63.6**†	5.0
	Trg-I	-	75.6	**76.7**†	1.1
	Trg-C	-	72.8	**73.3**†	0.5
	Arg-I	-	57.3	**59.5**†	2.2
	Arg-C	-	54.8	**57.5**†	2.7
ERE-EN	Ent-C	-	87.0	**87.4**	0.4
	Rel-C	-	53.2	**56.1**†	2.9
	Trg-I	-	68.4	**69.3**†	0.9
	Trg-C	-	57.0	**57.9**†	0.9
	Arg-I	-	50.1	**52.2**†	2.1
	Arg-C	-	46.5	**48.6**†	2.1

Table 7. F1 scores of the models on the test data of English datasets. Δ indicates the performance difference between FourIE and OneIE. Rows with † designate the significant improvement ($p < 0.01$) of FourIE over OneIE.

As can be seen from the table, FourIE is consistently better than the two baseline models (DyGIE++ and OneIE) across different datasets and tasks. The performance improvement is significant for almost all the cases and clearly demonstrates the effectiveness of the proposed model.

Finally, Table 8 reports the performance of FourIE and OneIE on the Chinese and Spanish datasets (i.e., ACE05-CN and ERE-ES). In addition to the monolingual setting (i.e., trained and evaluated on the same languages), following (Y. Lin et al., 2020a), we also evaluate the models on the multilingual training settings where ACE05-CN and ERE-ES are combined with their corresponding English datasets ACE05-E+ and EAE-EN (respectively) to train the models (for the four IE tasks), and the performance is then evaluated on the test sets of the corresponding languages (i.e., ACE05-CN and ERE-ES). It is clear from the table that FourIE also significantly outperforms OneIE across nearly all the different setting combinations for languages, datasets and tasks. This further illustrates the portability of FourIE to different languages.

Test Data	**Train Data**	**Task**	**OneIE**	**FourIE**	Δ%
ACE05-CN	ACE05-CN	Ent-C	88.5	**88.7**	0.2
		Rel-C	62.4	**65.1**†	2.7
		Trg-C	65.6	**66.5**†	0.9
		Arg-C	52.0	**54.9**†	2.9
	ACE05-CN ACE05-E+	Ent-C	**89.8**	89.1	-0.7
		Rel-C	62.9	**65.9**†	3.0
		Trg-C	67.7	**70.3**†	2.6
		Arg-C	53.2	**56.1**†	2.9
ERE-ES	ERE-ES	Ent-C	81.3	**82.2**†	0.9
		Rel-C	48.1	**57.9**†	9.8
		Trg-C	56.8	**57.1**	0.3
		Arg-C	40.3	**42.3**†	2.0
	ERE-ES ERE-EN	Ent-C	81.8	**82.7**†	0.9
		Rel-C	52.9	**59.1**†	6.2
		Trg-C	59.1	**61.3**†	2.2
		Arg-C	42.3	**45.4**†	3.1

Table 8. F1 scores on Chinese and Spanish test sets. † marks the significant improvement ($p < 0.01$) of FourIE over OneIE.

Effects of $\texttt{GCN}^{inst}$ and $\texttt{GCN}^{type}$. This section evaluates the contributions of the two important components in our proposed model FourIE, i.e., the instance interaction graph with $\texttt{GCN}^{inst}$ and the type dependency graph with $\texttt{GCN}^{type}$. In particular, we examine the following ablated/varied models for FourIE: (i) "**FourIE**-$\texttt{GCN}^{inst}$": this model excludes the instance interaction graph and the GCN model $\texttt{GCN}^{inst}$ from FourIE so the initial instance representations $\mathbf{r}_k$ are directly used to predict the types for the instances (replacing the enriched vectors $\mathbf{r}_k^{inst}$), (ii) "**FourIE**-$\texttt{GCN}^{type}$": this model eliminates the type dependency graph and the GCN model $\texttt{GCN}^{type}$ (thus the loss term L_{dep} as well) from FourIE, (iii) "**FourIE**-$\texttt{GCN}^{inst}$-$\texttt{GCN}^{type}$": this model removes both the instance interaction and type dependency graphs from FourIE, (iv) "**FourIE**-$\texttt{GCN}^{type}$+TDDecode": this model also excludes $\texttt{GCN}^{type}$; however, it additionally applies the global type dependencies features to score the joint predictions for the beam search in the decoding step (the implementation for this beam search is inherited from (Y. Lin et al., 2020a) for a fair comparison), and (v) "**FourIE**-$\hat{\mathbf{A}}^{pred}$": instead of employing the approximation matrix $\hat{\mathbf{A}}^{pred}$ in FourIE, this model directly uses the adjacency matrix $\mathbf{A}^{pred}$ in the L_{dep} regularizer (L_{dep} thus does not influence the instance representation-related parameters θ^{inst}). Table 9 shows the performance of the models on the development dataset of ACE05-E+ for four IE tasks.

Models	Ent-C	Rel-C	Trg-C	Arg-C
FourIE	**89.6**	**64.3**	**71.0**	**59.0**
FourIE-$\texttt{GCN}^{inst}$	89.1	62.3	70.3	57.5
FourIE-$\texttt{GCN}^{type}$	88.5	61.8	69.9	56.6
FourIE-$\texttt{GCN}^{inst}$-$\texttt{GCN}^{type}$	88.2	59.3	68.9	56.1
FourIE-$\texttt{GCN}^{type}$+TDDecode	88.8	59.6	70.8	56.8
FourIE-$\hat{\mathbf{A}}^{pred}$	89.0	62.3	70.2	57.6

Table 9. F1 scores of the models on the ACE05-E+ dev data.

The most important observation from the table is that both $\texttt{GCN}^{inst}$ and $\texttt{GCN}^{type}$ are necessary for FourIE to achieve the highest performance for the four IE tasks. Importantly, replacing $\texttt{GCN}^{type}$ in FourIE with the global type dependency features for decoding (i.e., "**FourIE**-$\texttt{GCN}^{type}$+TDDecode") as in (Y. Lin et al., 2020a) or eliminating the approximation $\hat{\mathbf{A}}^{pred}$ for L_{dep} produces inferior performance, especially for relation and argument extraction. This clearly demonstrates the benefits for deeply integrating knowledge from type dependencies to influence representation learning parameters with L_{dep} for joint four-task IE.

Contributions of Type Dependency Edges. Our type dependency graphs $\mathbf{G}^{gold}$ and $\mathbf{G}^{pred}$ involves three categories of edges, i.e., entity_relation, entity_argument, and event_argument type edges. Table 10 presents the performance of FourIE (on the development data of ACE05-E+) when each of these edge categories is excluded from our type dependency graph construction.

Models	**Ent-C**	**Rel-C**	**Trg-C**	**Arg-C**
FourIE	**89.6**	**64.3**	**71.0**	**59.0**
FourIE - entity_relation	88.7	61.9	71.0	57.5
FourIE - entity_argument	89.3	63.2	70.0	56.9
FourIE - event_argument	89.5	64.1	69.8	57.7

Table 10. F1 scores of the ablated models for type dependency edges on the ACE05-E+ dev data.

The table clearly shows the importance of different categories of type dependency edges for FourIE as the elimination of any category would generally hurt the performance of the model. In addition, we see that the contribution level of the type dependency edges intuitively varies according to the tasks of consideration. For instance, entity_relation type edges are helpful mainly for entity mention, relation and argument extraction. Finally, an error analysis is conducted in the appendix to provide insights about the benefits of the type dependency graphs $\mathbf{G}^{gold}$ and $\mathbf{G}^{pred}$ for FourIE (i.e., by comparing the outputs of FourIE and "**FourIE**-$\texttt{GCN}^{type}$").

3.1.5 Related Work. The early joint methods for IE have employed feature engineering to capture the dependencies between IE tasks, including Integer Linear Programming for Global Constraints (Q. Li, Anzaroot, Lin, Li, & Ji, 2011; Roth & Yih, 2004a), Markov Logic Networks (Riedel, Chun, Takagi, & Tsujii, 2009; Venugopal, Chen, Gogate, & Ng, 2014), Structured Perceptron (Judea & Strube, 2016; Q. Li, Ji, Hong, & Li, 2014; Q. Li et al., 2013a; Miwa & Sasaki, 2014), and Graphical Models (B. Yang & Mitchell, 2016a; Yu & Lam, 2010a).

Recently, the application of deep learning has facilitated the joint modeling for IE via shared parameter mechanisms across tasks. These joint models have focused on different subsets of the IE tasks, including EME and RE (Bekoulis, Deleu, Demeester, & Develder, 2018a; T.-J. Fu, Li, & Ma, 2019; Katiyar &

Cardie, 2017; Luan et al., 2019a; C. Sun et al., 2019; Veyseh, Dernoncourt, Dou, & Nguyen, 2020a; Veyseh, Dernoncourt, Thai, Dou, & Nguyen, 2020a; Zheng et al., 2017a), event and temporal RE (Han, Ning, & Peng, 2019), and ETD and EAE (T. H. Nguyen, Cho, & Grishman, 2016a; T. M. Nguyen & Nguyen, 2019a; Zhang et al., 2019). However, none of these work has explored joint inference for four IE tasks EME, ETD, RE, and EAE as we do. The two most related works to ours include (Wadden et al., 2019a) that leverages the BERT-based information propagation via dynamic span graphs, and (Y. Lin et al., 2020a) that exploits BERT and global type dependency features to constrain the decoding step. Our model is different from these works in that we introduce a novel interaction graph for instance representations for four IE tasks and a global type dependency graph to directly inject the knowledge into the training process.

3.1.6 Summary.

We present a novel deep learning framework to jointly solve four IE tasks (EME, ETD, RE, and EAE). Our model attempts to capture the inter-dependencies between instances of the four tasks and their types based on instance interaction and type dependency graphs. GCN models are employed to induce representation vectors to perform type predictions for task instances and regularize the training process. The experiments demonstrate the effectiveness of the proposed model, leading to SOTA performance over multiple datasets on English, Chinese, and Spanish.

3.2 DepIE

3.2.1 Introduction.

Entity mention recognition (EMR), event trigger detection (ETD), event argument extraction (EAE), and relation extraction (RE) are four main challenging tasks in information extraction (IE), which aim to extract entities (e.g., a person), events (e.g., an attack), event arguments (e.g., a victim in an attack), and relations (e.g., work-for) mentioned in text. These IE tasks have been solved mostly in pipelined approaches (Y. Chen, Xu, Liu, Zeng, & Zhao, 2015c; Du & Cardie, 2020; V. D. Lai et al., 2020; F. Li et al., 2020b; Q. Li, Ji, & Huang, 2013b; M. V. Nguyen, Nguyen, et al., 2021; T. H. Nguyen & Grishman, 2015a; Pouran Ben Veyseh, Lai, Dernoncourt, & Nguyen, 2021; Veyseh, Nguyen, & Nguyen, 2020a), where input to a model performing an IE task involves predictions from other models performing other IE tasks. As a result, errors in predictions by a model can be propagated to subsequent models in the pipeline to hurt overall performance.

To avoid error propagation, the four IE tasks can be solved jointly (JointIE) in a single model (Y. Lin et al., 2020b; M. V. Nguyen, Lai, & Nguyen, 2021; Zhang & Ji, 2021b). As such, a key challenge for JointIE models is to effectively capture dependencies between the IE tasks to boost overall extraction performance. In particular, two types of task dependencies are important for JointIE, i.e., cross-instance and cross-type dependencies. First, for cross-instance dependencies, JointIE models use instances to refer to word spans for event triggers/entity mentions (for EMR and ETD) or pair of word spans of event triggers/entity mentions (for EAE and RE) that should be classified according to predefined information types for IE. Accordingly, an important insight from previous JointIE models is to enrich the representation for one instance with those from related instances in different IE tasks to facilitate the type prediction (Y. Lin et al., 2020b; M. V. Nguyen, Lai, & Nguyen, 2021). To this end, a typical approach to encode cross-instance dependencies for representation learning in previous work involves creating dependency graphs between instances to connect related instances to facilitate representation learning (M. V. Nguyen, Lai, & Nguyen, 2021; Zhang &

Ji, 2021b). However, as the instance dependency graphs in previous work are only created manually using some heuristics, e.g., connecting instances that share an entity mention or event trigger (M. V. Nguyen, Lai, & Nguyen, 2021), they might be suboptimal for a given dataset and hinder further performance improvement for IE.

Consequently, to improve representation enrichment with information from related instances for JointIE, our work proposes to automatically learn cross-instance dependency graphs for IE tasks from data. To enable maximal flexibility, we explore a fully connected graph between all task instances in a sentence where a dependency weight is assigned to each edge to quantify the relatedness between two instances. In our method, we argue that dependency weights between task instances should be computed over multiple sources of information to produce optimal and comprehensive dependency graphs. To this end, motivated by the encoding of different linguistic structures (e.g., semantics, syntax) in the layers of pre-trained language models (PLMs), e.g., BERT (Devlin et al., 2019b; Jawahar, Sagot, & Seddah, 2019), we propose to leverage the representations of instances at different layers of PLMs to compute dependency weights for the instances. In particular, given two instances for JointIE, their representation vectors at each layer of a PLM are consumed to produce a layer-specific dependency weight, which will be combined across layers to obtain an overall weight for our dependency graph. Graph Convolutional Networks (GCNs) (Kipf & Welling, 2017; T. H. Nguyen & Grishman, 2018a) will then be used to induce enriched representations for the instances based on the computed cross-instance dependency graph.

In addition, cross-type dependencies/patterns in JointIE systems characterize co-occurrences/co-relations of information types of different IE tasks (e.g., entity/event types and argument roles) in a single input sentence. For instance, in the ACE 2005 dataset (Walker et al., 2006), a *"Victim"* argument for an *"Attack"* event is likely to be the *"Victim"* argument for a *"Die"* event in the same sentence. Accordingly, previous JointIE models have leveraged cross-type dependencies either in the decoding phase, i.e., to form global type patterns/graphs to constrain the type prediction (Y. Lin et al., 2020b), or in the training phase, i.e., to form type dependency graphs to aid consistency regularization of golden and predicted types (M. V. Nguyen, Lai, & Nguyen, 2021). However, as in cross-instance dependencies, the dependency graphs between information types in IE in previous work are also designed manually, e.g., by linking types that are involved in the same instance for some IE task (M. V. Nguyen, Lai, & Nguyen, 2021). This is not desirable as manual designs might miss important cross-type patterns that cannot guarantee optimal performance for JointIE.

To this end, we propose to further learn cross-type dependencies/patterns from data to better support type predictions of JointIE instances. As such, we view each information type in our IE tasks as a binary random variable, which is 1 if the type appears in the sentence, and 0 otherwise. This formulation enables us to employ Bayesian structure learning algorithms to infer dependency structures from data. In particular, we propose to leverage the Chow-Liu algorithm (Chow & Liu, 1968) that measures mutual information between any two types (variables) in training data to learn a first-order dependency tree, aiming to approximate the underlying joint distribution of the information types (types) for JointIE. Afterward, the resulting Chow-Liu tree containing induced dependencies between information types will be used to generate global cross-type patterns for JointIE.

To incorporate the learned cross-type dependencies into the JointIE model, our goal is to leverage such global patterns to obtain additional features to further enrich the GCN-induced representations for type prediction. Our intuition is to treat the induced cross-type patterns as anchor knowledge to which the information types, representations, and dependencies of IE instances in a sentence should adhere to exploit consistency and improve predictions for JointIE in the data. To this end, for each learned cross-type pattern, we seek to compute a similarity score between the computed cross-instance dependency graph for an input sentence and the cross-type pattern that can be included into the representations for the instances to predict types. Accordingly, we propose to leverage random walk graph kernels (Feng, You, Wang, & Tassiulas, 2022; Gärtner, Flach, & Wrobel, 2003) that facilitate similarity computation between two graphs (i.e., the cross-instance dependency graph and cross-type pattern) via counting common random walks on the graphs to enrich representations for JointIE. Finally, we evaluate the proposed model with induced cross-task and cross-type dependencies for JointIE in both monolingual and cross-lingual learning settings. Experimental results show that our model consistently outperforms strong baselines in all the settings across four different datasets and languages.

3.2.2 Model. There are four tasks in our IE pipeline, i.e., entity mention recognition (EMR), event detection (ED), event argument extraction (EAE), and relation extraction (RE). EMR and ED seek to identify word spans and types for entities (e.g., a *"Person"*) and events (e.g., an *"Attack"*) in text, respectively. On the other hand, EAE aims to identify whether each entity mention plays an argument role (e.g., an *"Attacker"*) in a given event mention. A special type *"Other-role"* is used to indicate that an entity does not play any role in a

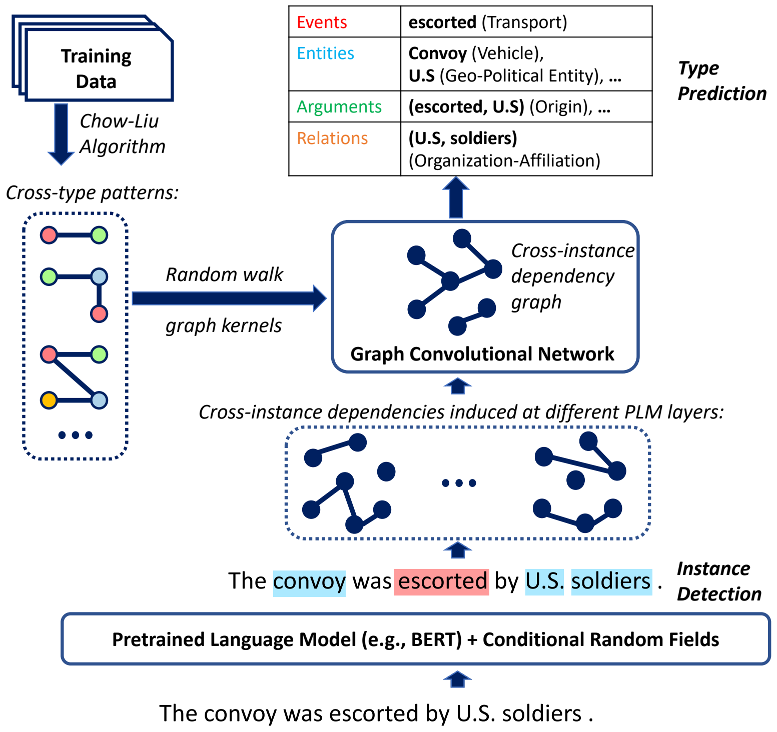

Training Data
Chow-Liu Algorithm
Cross-type patterns:
Random walk
graph kernels
Events
escorted (Transport)
Entities
Convoy (Vehicle), U.S (Geo-Political Entity), …
Arguments
(escorted, U.S) (Origin), …
Relations
(U.S, soldiers) (Organization-Affiliation)
Type Prediction
Cross-instance dependency graph
Graph Convolutional Network
Cross-instance dependencies induced at different PLM layers:
The convoy was escorted by U.S. soldiers .
Instance Detection
Pretrained Language Model (e.g., BERT) + Conditional Random Fields
The convoy was escorted by U.S. soldiers .

Figure 11. Overview of our JointIE model.

given event. For RE, the task is to determine if a relation (e.g., an *"Affiliation"* relation) exists between two given entity mentions. Similar to EAE, an special type *"Other-relation"* is used in RE to indicate no relation between two given entities. Joint information extraction (JointIE) is the joint task of EMR, ED, EAE, and RE (Y. Lin et al., 2020b; M. V. Nguyen, Lai, & Nguyen, 2021; Zhang & Ji, 2021b), which aims to simultaneously predict entity mentions, event triggers, event arguments and relations for an input text in an end-to-end fashion.

Our proposed model (called *"DepIE"*) for JointIE consists of three main components: (i) Instance Detection, (ii) Cross-Instance Dependencies, and (iii) Cross-type Dependencies. Figure 11 presents an overview for our model.

3.2.2.1 Instance Detection. The first step in our model is to identify candidate instances for all the four IE tasks. In particular, candidate instances for

EMR and ED involve spans of words for entity mentions and event triggers in text. For EAE, a candidate instance is formed by a pair of an event trigger span and an entity mention span. Similarly, we can obtain candidate instances for RE by pairing entity mention spans. Note that this step only performs candidate instance identification. Information types for the instances will be predicted in the next steps.

Event Triggers and Entity Mentions: Given an input sentence $\mathbf{w} = [w_1, \ldots, w_N]$ with N words, we employ a pretrained language model (PLM), e.g., RoBERTa (Y. Liu et al., 2019), to produce a sequence of contextualized embeddings $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$ for the words (using average of hidden vectors for word-pieces in the last layer of the PLM). The vector sequence $\mathbf{X}$ is then consumed by two different conditional random fields (CRFs) layers to predict two BIO tag sequences; each sequence aims to captures spans of event triggers (or entity mentions) for ED (or EMR). The negative log-likelihoods L_t and L_e returned by the CRFs for the ground-truth tag sequences of the spans for EMR and ED will then be included into the overall loss function. At test time, Viterbi algorithm is used to search for most probable tag sequences to find spans for event triggers $V_t = \{v_t\}$ and entity mentions $V_e = \{v_e\}$ (i.e., candidate instances) in the sentence. Each event trigger/entity mention is represented by a vector $\mathbf{v}_*$ ($* \in \{t, e\}$), computed via the average of contextualized embeddings for the words inside its corresponding spans v_*.

Event Arguments and Relations: While it is possible to use all pairs of entity mention and event trigger spans for the candidate instances of EAE and RE for type prediction, the large number of possible pairs will increase necessary computational resources. To this end, we first send the pairs into binary classifiers to determine if they are positive examples (i.e., corresponding to some actual types of interest for EAE and RE). In particular, to decide if an entity mention $v_e \in V_e$ plays any role with an event trigger $v_t \in V_t$, we concatenate their span vectors (i.e., $\mathbf{v}_e$ and $\mathbf{v}_t$) and feed the concatenation into a feed-forward network (FFN) with a sigmoid function in the end: $p_a = \sigma(FFN_a([\mathbf{v}_e; \mathbf{v}_t]))$. Here, the score $p_a \in (0, 1)$ represents the likelihood for v_e to be an argument of some role for v_t. Similarly, we can compute a score $p_r \in (0, 1)$ for all pairs of entity mentions $v_{e_1}, v_{e_2} \in V_e$ to estimate the likelihood that there exists a relation between the entity mentions. In the training process, we obtain the binary cross-entropy losses L_a and L_r computed with the probability scores p_a, p_r to include in the overall loss function. In test time, we employ a threshold of 0.5 for the scores p_a, p_r to determine positive pairs $V_a = \{v_a = (v_t, v_e)\}$ for event arguments and $V_r = \{v_r = (v_{e_1}, v_{e_2})\}$ for relations. Only positive pairs are retained for our next steps of type prediction. Finally, each positive event argument/relation is also represented by the average of representations of the involving event trigger and entity mention instances, called $\mathbf{v}_a$ and $\mathbf{v}_r$.

3.2.2.2 Cross-Instance Dependencies. Given the detected instances for the four IE tasks in $\mathbf{w}$, we aim to enrich the representation for each instance with information from other related instances to facilitate type prediction. As such, our model first learns a dependency graph $G^{inst} = (V, E)$ to capture the relatedness for the instances (called cross-instance dependency graph). In particular, the node set V of G^{inst} involves all the detected instances, i.e., $V = V_t \cup V_e \cup V_a \cup V_r$. To enable information flow across different instances, our edge set E will include an edge for each possible pair of instances in V; a weight α_{ij}

will be assigned to each pair (v_i, v_j) to quantify the dependency between v_i and v_j in V.

To learn the dependency weights α_{ij}, our intuition is to exploit information from different sources (e.g., semantics, syntax) to ensure comprehensive coverage of relatedness aspects for JointIE. Motivated by different linguistic features encoded in different transformer layers of PLMs (Jawahar et al., 2019), we propose to treat each layer of BERT (with L layers) as a source of information. In particular, each word in the input sentence will be represented by L different embeddings returned by each layer of the PLM. In this way, for each node in V, we can obtain L different node representations computed at each layer of BERT (by averaging representations for word-pieces). Let $\mathbf{v}_i^l, \mathbf{v}_j^l$ be the representations for the nodes $v_i, v_j \in V$ at layer l of the PLM. The dependency weight $\alpha_{ij}^l \in (0, 1)$ between the instance nodes v_i, v_j at layer l of BERT is computed by: $\alpha_{ij}^l = FFN_\sigma^l([\mathbf{v}_i^l; \mathbf{v}_j^l])$, where FFN_σ^l is a feed-forward network with a sigmoid function in the end.

To this end, each instance $v_i \in V$ is associated with L sets of weights $\{\alpha_{ij}^l\}$ capturing its dependencies on the other instances according to L different sources of information from BERT. The importance of the l-th information source to representation learning of v_i is then measured by sending its l-th representation $\mathbf{v}_i^l$ to a feed-forward network $FFN_{src}(\mathbf{v}_i^l)$. Afterward, we normalize the layer-specific importance scores for v_i across layers with softmax, leading to $s_i^l = \text{softmax}_l(FFN_{src}(\mathbf{v}_i^{1:L}))$. The dependency weight between v_i and v_j in our cross-instance graph is then determined via: $\alpha_{ij} = \sum_l s_i^l \alpha_{ij}^l$.

Finally, the induced dependency graph with weights α_{ij} is used to enhance the representations for $v_i \in V$ via a Graph Convolutional Network (GCN) (Kipf &

Welling, 2017; T. H. Nguyen & Grishman, 2018a) with K layers:

$$\mathbf{h}_i^k = \text{ReLU}(\frac{\sum_{v_j \in V} \alpha_{ij} \mathbf{W}^k \mathbf{h}_j^{k-1} + \mathbf{b}^k}{\sum_{v_j \in V} \alpha_{ij}}), 1 \leq k \leq K$$

where $\mathbf{h}_i^k$ is the representation for v_i at the k-th layer of GCN ($\mathbf{h}_i^0 = \mathbf{v}_i$). For convenience, let $\mathbf{h}_i$ be the representation for the instance v_i at the final layer of the GCN, i.e., $\mathbf{h}_i = \mathbf{h}_i^K$.

3.2.2.3 Cross-Type Dependencies. As discussed in the introduction, to further improve the representations for the instances v_i for type prediction, our method proposes to induce global dependencies between information types for different IE tasks (called cross-type dependencies) from data and use them as knowledge to generate additional features for instance representations.

Cross-type Dependency Induction: For convenience, let T be the set of all information types for our four IE tasks, i.e., including entity types, event types, event argument roles, and relations. To infer dependencies/patterns between the types in T, our goal is to leverage their co-occurrences in the sentences of training data for the computation. As such, we consider the information types in T as random variables and leverage the well-known Chow-Liu algorithm (Chow & Liu, 1968) in Bayesian structure learning to find meaningful relationships/patterns among the types. The Chow-Liu algorithm approximates the underlying joint distribution of random variables by finding a first-order dependency tree among the variables (i.e., tree nodes correspond to the variables).

Let $X_i \in \{0, 1\}$ be the binary random variable for the information type $t_i \in T$ where $X_i = 1$ if there exists one instance with type t_i in the current sentence, and $X_i = 0$ otherwise. The algorithm then computes mutual information (MI) scores between any two random variables X_i, X_j via:

$$I(X_i, X_j) = \sum_{x_i, x_j \in \{0,1\}} \hat{P}(x_i, x_j) \log \frac{\hat{P}(x_i, x_j)}{\hat{P}(x_i)\hat{P}(x_j)}$$

where $\hat{P}(x_i, x_j) = \frac{count(X_i = x_i, X_j = x_j)}{M}$ is the empirical joint distribution between X_i and X_j computed by counting across training data (M is the total number of sentences in the training data). Similarly, we can compute the marginal distributions $\hat{P}(x_i)$ and $\hat{P}(x_j)$. Afterwards, we construct a cross-type dependency tree G^{ctp} for information types as the spanning tree over the random variables that achieves maximum sum of the MI scores. The maximum spanning tree can be solved via Kruskal (Kruskal, 1956) or Prim (Prim, 1957) algorithms.

To make it more manageable, we collect the set of connected sub-graphs (i.e., trees) U that have at least two nodes and less than n nodes in G^{ctp} ($2 \leq n \leq |T|$ is a hyper-parameter) to serve as the global cross-type patterns/dependencies induced by our method for JointIE.

Feature Generation with Graph Kernels: Using the induced cross-type patterns $G_d^{ctp} \in U$ from data as anchor knowledge, we expect the information types, instance representations, and instance dependencies in an input sentence $\mathbf{w}$ to follow the patterns to exploit consistency in the data. In particular, instance representations and dependencies in an input sentence will have higher quality for type prediction if they are more similar to the induced cross-type patterns from data. Accordingly, we propose to leverage similarity scores between the cross-instance dependency graph for $\mathbf{w}$ and the cross-type patterns in U as additional features to improve representations for JointIE. Here, we can employ the cross-instance dependency graph G^{inst} with dependency weights α_{ij} computed in the previous step for the feature computation.

As such, to compute the similarity between G^{inst} and G_d^{ctp}, we propose to employ random walk graph kernels (Gärtner et al., 2003) that can facilitate similarity measurement between two graphs with different number of nodes. In particular, the random walk kernel is computed by counting the number of common random walks on the two graphs, which has been shown to be equivalent to performing a random walk on the direct product of the graphs (Vishwanathan, Borgwardt, Schraudolph, et al., 2006). This enables the p-step random walk kernel between two graphs G_1 and G_2 to be efficiently computed via: (Feng et al., 2022; Vishwanathan et al., 2006):

$$K_p(G_1, G_2) = \sum_{i,j} \left[(\mathbf{V}_1 \mathbf{V}_2^T) \odot (\mathbf{A}_1^p \mathbf{V}_1 (\mathbf{A}_2^p \mathbf{V}_2)^T) \right]_{ij}$$

where $\mathbf{V}_1$ and $\mathbf{V}_2$ are the node embedding matrices for the node sets; $\mathbf{A}_1$ and $\mathbf{A}_2$ are adjacency matrices for the graphs G_1 and G_2 respectively; $\odot$ is the element-wise product, and $\mathbf{A}_*^p$ is the p-th power of the matrix $\mathbf{A}_*$ ($* \in \{1, 2\}$).

To adapt this random walk kernel for G^{inst} and G_d^{ctp}, we can obtain the adjacency matrix $\mathbf{A}^{inst}$ for G^{inst} from the dependency weights α_{ij}, i.e., $\mathbf{A}_{ij}^{inst} = \alpha_{ij}$. The node embedding matrix $\mathbf{V}^{inst}$ for G^{inst} can leverage the GCN-induced vectors by setting the i-th row of $\mathbf{V}^{inst}$ to $\mathbf{h}_i$ for instance $v_i \in V$. Also, for each induced cross-type pattern/tree $G_d^{ctp} \in U$, we can use its binary adjacency matrix $\mathbf{A}_d^{ctp}$ for the kernel computation. Its node embedding matrix $\mathbf{V}_d^{ctp}$ will be produced by looking up the corresponding types in a type embedding matrix $\mathbf{T}$ for all types in T. In our method, $\mathbf{T}$ is initialized randomly so its embedding dimension is equal to those for the instance representation $\mathbf{h}_i$. In this way, we can compute a kernel-based similarity score $ks_d = K_p(G^{inst}, G_d^{ctp})$ between the cross-instance dependency graph G^{inst} and each cross-type pattern in U. Finally, the concatenation of such similarity scores, i.e., $\mathbf{m}^{ctp} = [ks_1, ks_2, \ldots, ks_{|U|}]$, can be used to provide additional

Model	**ACE05-E+ (English)**				**ACE05-CN (Chinese)**				**ACE05-AR (Arabic)**				**ERE-ES (Spanish)**			
	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg
Text2event	-	-	71.8	54.4	-	-	-	-	-	-	-	-	-	-	-	-
DEGREE-E2E	-	-	71.7	56.8	-	-	-	-	-	-	-	-	-	-	-	-
Query&Extract	-	-	73.6	55.1	-	-	-	-	-	-	-	-	-	-	-	-
GTEE-DYNPREF	-	-	74.3	54.7	-	-	-	-	-	-	-	-	-	-	-	-
OneIE	90.8	60.4	72.5	56.3	88.5	64.9	67.3	54.8	81.2	59.0	56.6	37.2	83.7	57.5	58.3	42.5
AMRIE	91.0	62.8	72.7	57.7	-	-	-	-	-	-	-	-	-	-	-	-
FourIE	91.1	63.1	72.8	58.3	88.8	66.0	69.1	57.5	81.7	61.4	57.9	42.1	83.8	59.0	63.4	45.1
DepIE (Ours)	91.7	64.9	**74.6**	61.2	**89.2**	68.3	74.3	60.0	82.7	63.5	63.1	46.4	86.5	61.2	65.9	51.9

Table 11. Monolingual performance on test data of the datasets. *"Ent"*, *"Rel"*, *"Trg"*, and *"Arg"* indicate F1 scores for identification and classification of entity mentions, relations, event triggers, and arguments respectively. All results are reported by the original papers or produced by running the official code. All JointIE models use large RoBERTa. Underlined numbers indicate that *DepIE* is significantly better than the baselines ($p < 0.01$).

global features for the instance representations for type predictions. Note that in this way, our cross-type patterns can support both training and test phases for JointIE models. This is in contrast to previous methods that can only utilize manually designed patterns in either training (e.g., FourIE) or decoding (e.g., OneIE) phase.

Training: To predict type for each instance $v_i \in V$, we compute an overall representation vector $\mathbf{r}_i$ for v_i by concatenating its GCN-induced representation $\mathbf{h}_i$ and the global features $\mathbf{m}^{ptn}$: $\mathbf{r}_i = FFN_{pred}(\text{concat}(\mathbf{h}_i, \mathbf{m}^{ptn}))$. Here, FFN_{pred} is a feed-forward network to ensure that $\mathbf{r}_i$ has the same dimension as the type embeddings $\mathbf{T}$. The type distribution v_i is then estimated by normalizing the similarity of $\mathbf{r}_i$ and the type embeddings: $\hat{\mathbf{y}}_i = \text{softmax}(\mathbf{r}_i\mathbf{t}^T|\mathbf{t} \in \mathbf{T}_i)$ where $\mathbf{T}_i$ is the set of embeddings for all possible types T_i for v_i in T. The negative log-likelihood of the ground-truth types t_i is then used to train our model: $L_{cls} = -\sum_{v_i \in V} \log(\hat{\mathbf{y}}_i[t_i])$. In summary, the overall training loss for our model is: $L = L_t + L_e + L_a + L_r + L_{cls}$.

3.2.3 Experiments. **Datasets:** Following previous work (Y. Lin et al., 2020b; M. V. Nguyen, Lai, & Nguyen, 2021), we conduct experiments on four datasets with different languages, i.e., ACE05-E+ (English), ACE05-CN (Chinese), ACE05-AR (Arabic), and ERE-ES (Spanish). The three ACE05 datasets are created by the Automatic Content Extraction program (Walker et al., 2006) with 33 event types, 7 entity types, 6 relation types, and 22 argument roles; and the ERE-ES dataset is from the Deep Exploration and Filtering of Text program (DEFT) (Song et al., 2015) with a similar schema to ACE05 datasets. For a fair comparison, we use the same preprocessing and train/dev/test splits for ACE05-E+, ACE05-CN, and ERE-ES as provided by prior work (Y. Lin et al., 2020b; M. V. Nguyen, Lai, & Nguyen, 2021). The ACE05-AR dataset does not have a standard split for JointIE so we follow the data split by (M'hamdi et al., 2019) for ETD in Arabic and apply the same preprocessing code from previous work (Y. Lin et al., 2020b) to produce the train/dev/test sets for ACE05-AR. Additionally, we perform experiments on the IARPA BETTER program[3]'s Basic Event Extraction datasets, which feature 118 event types, 3 mention types, and 3 argument roles. The BETTER-EN dataset is obtained by respectively combining the official training, development, and test parts of Phase 1, 2, and 3 English data. For the BETTER-FA dataset, we randomly split the Phase 2 Farsi evaluation data into training, development, and test portions with a ratio of 70/15/15 as no standard split is provided. Statistics for all the datasets are shown in Table 12.

Hyper-Parameters: For the PLMs, we use RoBERTa large (Y. Liu et al., 2019) and its multilingual version XLM-RoBERTa large (Conneau et al., 2020) for English and non-English datasets respectively. We tune hyper-parameters for our model on ACE05-E+ development data and apply the best hyper-parameters to the other datasets for consistency. In particular, we select: $5e$-6 for learning rate

[3]https://www.iarpa.gov/index.php/research-programs/better

Datasets	**Split**	**#sents**	**#ents**	**#rels**	**#events**
ACE05-E+	Train	19,240	47,525	7,152	4,419
	Dev	902	3,422	728	468
	Test	676	3,673	802	424
ACE05-CN	Train	6,841	29,657	7,934	2,926
	Dev	526	2,250	596	217
	Test	547	2,388	672	190
ACE05-AR	Train	1,915	28,113	4,063	1,198
	Dev	108	1,892	275	112
	Test	152	2,495	374	169
ERE-ES	Train	7,067	11,839	1,698	3,272
	Dev	556	886	120	210
	Test	546	811	108	269
BETTER-EN	Train	5,617	18,815	-	16,594
	Dev	1,163	3,958	-	3,177
	Test	1,173	3,707	-	3,311
BETTER-FA	Train	2,932	11,612	-	10,100
	Dev	592	2,377	-	2,061
	Test	658	2,468	-	2,054

Table 12. Dataset statistics. **#sents**, **#ent**, **#rels**, and **#events** represent the numbers of sentences, entity mentions, relations, and events respectively.

with Adam optimizer; 10 for batch size; 300 for the hidden vector sizes for all the feed-forward networks and the GCN model; 2 for the number of layers for the feed-forward and GCN networks; $n = 4$ for the sizes of cross-type patterns in U; and $p = 2$ for the kernel computation. The model performance is obtained by averaging over three runs with different random seeds.

Baselines: We compare our method (i.e., **DepIE**) with recent models that jointly perform our four IE tasks, including **OneIE** (Y. Lin et al., 2020b), **AMRIE** (Zhang & Ji, 2021b), and **FourIE** (M. V. Nguyen, Lai, & Nguyen, 2021). FourIE is the current state-of-the-art method for JointIE. Among models, OneIE, FourIE, and our model DepIE are language-agnostic so they can be directly applied to non-English datasets. In contrast, AMRIE is only designed for English as it requires an English AMR parser. To be comprehensive, we also consider recent event extraction methods, i.e., **Text2event** (Lu et al., 2021b), **DEGREE-E2E** (I. Hsu et al., 2021), **Query&Extract** (S. Wang, Yu, Chang, Sun, & Huang, 2022), **GTEE-**

DYNPREF (X. Liu, Huang, Shi, & Wang, 2022), which perform only ETD and EAE.

Datasets	Task	OneIE	FourIE	DepIE (Ours)
BETTER-EN (English)	Ent	75.1	75.3	**76.5**
	Trg	63.6	63.9	**65.6**
	Arg	62.4	64.5	**65.6**
BETTER-FA (Farsi)	Ent	65.1	65.7	**66.5**
	Trg	57.0	57.6	**59.1**
	Arg	55.2	56.3	**58.1**

Table 13. Monolingual performance (F1 scores) on test data of BETTER datasets.

Test Data	Task	OneIE	FourIE	DepIE (Ours)
ACE05-CN	Ent	70.2	70.8	**71.8**
	Rel	31.1	32.6	**35.7**
	Trg	58.4	60.5	**62.1**
	Arg	37.9	39.2	**41.5**
ACE05-AR	Ent	64.2	65.4	**66.5**
	Rel	27.1	30.6	**31.7**
	Trg	35.4	36.9	**40.6**
	Arg	25.0	26.5	**28.0**
ERE-ES	Ent	75.5	76.5	**76.6**
	Rel	27.7	28.6	**33.0**
	Trg	45.3	47.0	**49.9**
	Arg	34.2	35.4	**37.4**
BETTER-FA	Ent	74.1	74.2	**74.8**
	Trg	56.5	57.3	**58.7**
	Arg	59.8	61.7	**63.0**

Table 14. Cross-lingual performance (F1 scores) on test data of non-English datasets. For the BETTER-FA setting, the models are trained on training data of BETTER-EN only. For the other settings, only training data of ACE05-E+ is used for training.

Monolingual Performance: We first compare the models in monolingual settings across the four datasets in Tables 11 and 13 where models are trained and tested on data of the same language. As can be seen, our model performs significantly better than the baselines across the datasets. Among the four IE tasks, the EAE and RE tasks appear to gain largest performance improvements. Further, as the improvements are consistent across languages, it highlights the portability to different languages of the induced cross-instance and cross-task dependencies in our proposed model for JointIE.

Crosslingual Performance: To further investigate the cross-lingual generalization of the JointIE models, we compare OneIE, FourIE, and DepIE in the cross-lingual transfer learning settings where the models are trained on training data of English datasets and evaluated on the test data of the other languages. As shown in Table 14, our model DepIE is still the best performer in the crosslingual settings over different tasks and test languages. The performance improvement is significant on almost all tasks ($p < 0.01$), thus demonstrating language-invariant advantages of our designed cross-task dependencies for JointIE. In addition, we note that this is the first comprehensive evaluation of JointIE models in cross-lingual transfer learning. As the performance of the current models is still not satisfactory, it emphasizes the challenges of JointIE with cross-lingual transfer learning and call for future research efforts in this important direction.

Models	**ACE05-E+**			
	Ent	Rel	Trg	Arg
DepIE	**89.1**	**65.6**	**73.3**	**65.3**
- *cross-instance*	87.4	62.7	71.7	62.0
+ *single-source graph*	88.6	64.3	72.7	63.7
+ *heuristic graph*	88.1	63.1	72.2	62.9
- *GCN*	88.3	63.8	72.4	63.1
- *cross-type*	88.2	64.1	72.0	64.1
+ *naive cross-type*	87.8	63.5	71.6	63.7
+ *cosine similarity*	88.4	64.5	72.8	64.3
+ *type regularization*	88.2	64.6	72.4	64.5
+ *global features*	87.7	63.1	72.0	64.0

Table 15. Model performance (F1) of ablated models.

Ablation Study: To study the impact of each proposed component for DepIE, Table 15 evaluates the ablated models over ACE05-E+ development data.

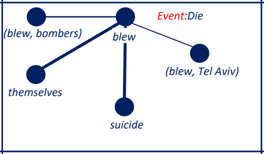
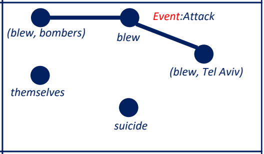
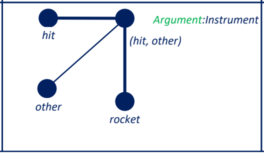
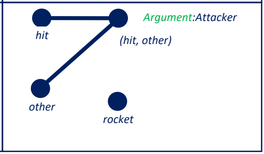
Example	DepIE	FourIE
In the January attack, two Palestinian suicide bombers blew themselves up in central Tel Aviv, killing 23 other people. ***Analysis**: DepIE can successfully predict "blew" as a "Die" event trigger due to the recognized connections with "suicide" and "themselves" while FourIE fails to do so.*		
A second rocket landed in farmlands and the other hit a house inside the refugee camp, … ***Analysis**: DepIE can successfully predict "other" as an "Instrument" for the event trigger "hit" due to its ability to connect to the important related instance "rocket" while FourIE fails to do so.*		

Figure 12. Some task instances along with their dependency connections produced by DepIE and FourIE.

In particular, for cross-instance dependencies, we first remove the cross-instance dependency graph from DepIE. The ablated model *"- cross-instance"* shows significant performance drops across all the four IE tasks, demonstrating the importance of the cross-instance dependency component to our model. In addition, we evaluate a simplified version of this component where a single source of information is used to induce dependencies between instances. Particularly, the cross-instance dependency weights α_{ij} in this case are computed with only the last layer of the PLM instead of all the layers. As the performance of the ablated model *"+single-source graph"* is substantially worse than the full model, it confirms the benefits of using multiple information sources from PLM to compute cross-instance dependencies for FourIE. Moreover, we replace our induced dependency weights for instances with the heuristic-based dependency weights produced by the best baseline model FourIE (i.e., $\alpha_{ij} = 1$ if instances v_i and v_j share an event trigger or entity mention). The inferior performance of the resulting model *"+heuristic graph"* compared to *"+single-source graph"* and DepIE strongly indicates the strength of automatically learned dependency graphs for JointIE. Finally, we report the performance of DepIE where the GCN model is removed while still preserving the cross-instance and cross-type dependencies (i.e., *"- GCN"*). As such, the contextualized embeddings $\mathbf{x}_i$ will replace the GCN-induced vectors $\mathbf{h}_i$ in the computation. It is clear from the table that the GCN model is necessary for DepIE as *"- GCN"* has significantly worse performance.

Next, we study the effect of the cross-type dependency component for DepIE. As shown in the table, removing cross-type dependencies from DepIE (i.e., *"- cross-type"*) significantly hurts model performance. To understand the benefit of the Chow-Liu algorithm, we examine a simpler method to produce the cross-type dependency graph G^{ctp} where two information types in T are connected if they are both expressed in a sentence in training data. The resulting model (i.e., *"+ naive cross-type"*) performs much poorer than our full model with the Chow-Liw tree. To investigate the effectiveness of the random walk kernels, we examine a similar method to the type dependency regularization in FourIE to compute the similarity between the cross-instance graph G^{inst} and the cross-type patterns G^{cpt}_d for the global features $\mathbf{m}^{cpt}$. In particular, we use a GCN model to consume the graphs G^{inst} and G^{cpt}_d along with their node embeddings; the resulting vectors for each graph are then max-pooled to obtain a representation vector for the graph. The similarity between the two graphs is then computed via the cosine similarity between their representations. As the corresponding model *"+ cosine similarity"* is worse than the full model over different tasks, it demonstrates the necessity of the random walk kernels for DepIE.

Finally, we remove the cross-type dependency component (i.e., with Chow-Liu and graph kernels) and integrate alternative methods to generate and apply cross-type dependencies from previous JointIE methods into DepIE, i.e., the type regularization in FourIE for training or the global type features for decoding in

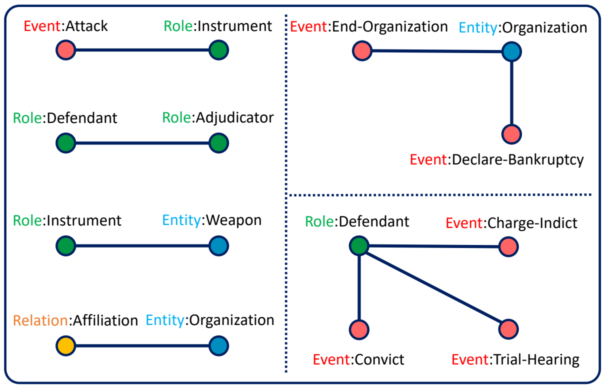


Figure 13. Cross-type patterns learned DepIE on ACE05-E+. Blue, red, green, and orange circles represent entity, event, argument role, and relation types respectively.

OneIE. Both the models *"+type regularization"* and *"+global features"* in Table 15 observe large decreased performance, further confirming the benefit of the cross-type dependency components for JointIE in DepIE.

Analysis: To understand the effect of the cross-instance dependency graph learned by DepIE compared to the heuristic-based dependency graph produced by FourIE, we examine examples on the ACE05-E+ development data for which DepIE can have correct predictions while FourIE fails to do so. Figure 12 presents some examples of this type. As can be seen, by computing dependency weights for all possible pairs of instances, DepIE can discover important related instances that do not share any entity mentions/event triggers with the instance of interest (e.g., the related instance *"suicide"* for *"blew"*), thus allow DepIE to correct the wrong predictions in FourIE to improve the performance.

Finally, Figure 13 presents some cross-type patterns learned DepIE. We observe that 3-node and 4-node patterns can capture subtle structures between information types for JointIE (e.g., the *"Charge-Indict"*, *"Convict"*, and *"Trial-Hearring"* event types and the *"Defendant"* argument role).

3.2.4 Related Work. IE tasks have been performed jointly to capture dependency between the tasks via feature engineering (Q. Li et al., 2013b; Roth & Yih, 2004b; B. Yang & Mitchell, 2016b; Yu & Lam, 2010b) or deep learning (Bekoulis, Deleu, Demeester, & Develder, 2018b; Luan et al., 2019b; T. H. Nguyen, Cho, & Grishman, 2016b; Zheng et al., 2017b) methods. However, most previous work only jointly solves two or three IE tasks (Lu et al., 2021b; T. M. Nguyen & Nguyen, 2019a). Recently, there have been growing interest in performing all the four IE tasks jointly (i.e., JointIE) (M. V. Nguyen, Min, et al., 2022a; Wadden, Wennberg, Luan, & Hajishirzi, 2019b; Zhang & Ji, 2021b) to exploit manually designed dependency graphs for IE instances (M. V. Nguyen, Lai, & Nguyen, 2021) or handcrafted global features for information types (Y. Lin et al., 2020b). Our work is different from previous JointIE models as we learn cross-instance and cross-type dependencies from data to provide better structures for representation learning. Finally, we note that our cross-type dependency component is related to structure learning methods for Bayesian networks (Banerjee & Ghosal, 2015; Eaton & Murphy, 2012; Scutari, Graafland, & Gutiérrez, 2019) and graph kernels to compute graph similarity (Feng et al., 2022; Gärtner et al., 2003; Kondor & Pan, 2016; Shervashidze, Vishwanathan, Petri, Mehlhorn, & Borgwardt, 2009; Vishwanathan et al., 2006). However, these approaches have not been explored for JointIE.

3.2.5 Summary. We present a novel model to jointly solve four IE tasks (EMR, ETD, EAE, and RE). Our model learns cross-instance dependencies through different layers of a PLM and cross-type dependencies via the Chow-Liu algorithm. The cross-task dependencies are exploited via GCNs and random walk kernels to improve representation learning. Extensive experiments demonstrate the state-of-the-art performance of our model across four datasets with different languages and settings.

3.3 GraphIE

3.3.1 Introduction. To extract structured information from unstructured text, a typical information extraction (IE) pipeline involves four major tasks: event trigger detection (ETD), event argument extraction (EAE), entity mention recognition (EMR), and relation extraction (RE). Previous work has performed such IE tasks via pipelined approaches (Y. Chen et al., 2015a; Du & Cardie, 2020; F. Li et al., 2020a; Q. Li et al., 2013a), where a model for one task uses output predictions from other models performing other tasks. Consequently, errors from the predictions can be propagated between the models in the pipeline.

Recently, ETD, EMR, EAE, and RE have been solved jointly in a single model, i.e., Joint Information Extraction - JointIE (Y. Lin et al., 2020a; M. V. Nguyen, Lai, & Nguyen, 2021; Wadden et al., 2019a; Zhang & Ji, 2021a), to avoid error propagation and leverage dependency between prediction instances of the four IE tasks (i.e., event trigger, entity mention, relation, and event argument candidates in a sentence). For example, if a *Person* entity mention is a *Victim* argument for a *Die* event, it is likely that the same entity mention is also a *Target* argument for an *Attack* event in the same sentence. To implicitly exploit instance dependency for representation learning, Wadden et al. (2019a) and Y. Lin et al. (2020a) employ a shared encoder to obtain representation vectors to classify instances of different IE tasks. Later work heuristically captures dependency between IE task instances via explicitly connecting the task instances that share an entity mention or event trigger (M. V. Nguyen, Lai, & Nguyen, 2021) or aligning the task instances that share text spans with some nodes on a semantic

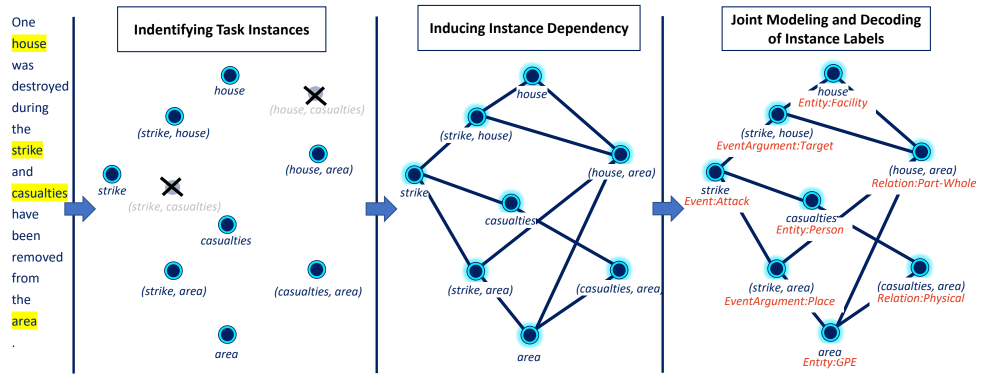


Figure 14. Overview of the three stages in our proposed model: i) identifying task instances, ii) inducing instance dependency, and iii) joint modeling and decoding of instance labels. Each node represents an instance for one of the four IE tasks, and edges (with weights ¿ 0.3) between nodes represent induced instance dependency.

graph (Zhang & Ji, 2021a) to aid representation learning. While natural, these manual designs for dependency between task instances might not be optimal for representation learning of JointIE.

In addition to representation learning, at the prediction level, previous work tends to factorize the joint distribution of labels for all the task instances in JointIE into the product of label distributions for each individual instance (i.e., performing local normalization), thus hindering the ability to fully exploit the interactions of instance labels across IE tasks. (Y. Lin et al., 2020a) and (Zhang & Ji, 2021a) mitigate this problem by decoding instance labels with handcrafted global features while (M. V. Nguyen, Lai, & Nguyen, 2021) focuses on encoding label interactions via consistency regularization over global type dependency graphs. However, these approaches still assume a factorization of the joint label distribution for prediction instances, thus unable to fundamentally address the label dependency encoding issue. Recently, some works have attempted to directly model the joint distribution of instance labels by reformulating JointIE tasks as text generation problems using state-of-the-art pre-trained seq2seq models, e.g., BART or T5 (Lewis et al., 2020;

Raffel et al., 2020a). In such generative models, text spans and labels for task instances are generated by the decoder in an autoregressive fashion to encode label dependency for joint distribution computation (I. Hsu et al., 2021; Lu et al., 2021a). Unfortunately, this approach needs to assume an order of the task instances to be decoded (e.g., from left to right) that disallows later instances in the order to interfere/correct predictions for earlier instances, causing suboptimal performance for JointIE.

In this work, we aim to overcome these issues by inducing dependency between the task instances for JointIE from data to boost representation learning, and directly modeling the joint distribution of the labels for all the task instances to fully enable label interactions. To this end, we consider each task instance as a node in a fully connected dependency graph; the weight for each edge is then learned to capture the dependency level between two corresponding instances. Note that this is different from prior work (M. V. Nguyen, Lai, & Nguyen, 2021; Zhang & Ji, 2021a) that heuristically designs sparser dependency graphs with disconnected task instance pairs, thus failing to explore all possible interactions between instance pairs for optimal representations. In our method, the induced dependency graph for instance nodes is then employed by Graph Convolutional Networks (GCNs) (Kipf & Welling, 2017; T. H. Nguyen & Grishman, 2018a) to enhance the representation for each instance node with information from all the other nodes according to their dependency levels. Afterwards, the enhanced instance representations and the induced dependency graph are utilized to estimate the joint distribution of instance labels via Conditional Random Fields (CRFs) (Lafferty, McCallum, & Pereira, 2001). This formulation enables us to approximately maximize the intractable joint likelihood of the ground-truth instance labels via Noise Contrastive Estimation (NCE) (Gutmann & Hyvärinen, 2012), which converts the maximization problem into the nonlinear logistic regression discriminating between the true labels and the noise labels.

Finally, previous work for JointIE has employed a greedy or beam search for decoding instance labels, which is not optimal due to their greedy nature. In this work, we propose a novel decoding algorithm for JointIE via Simulated Annealing (SA) (Kirkpatrick, Gelatt Jr, & Vecchi, 1983), which has been shown to be able to approximate the global optimum of a function (Kirkpatrick et al., 1983; Van Laarhoven & Aarts, 1987). Experimental results show that our proposed model for JointIE significantly outperforms previous models on multiple tasks with large margins across 5 datasets and 2 languages.

3.3.2 Problem Statement. Given an input sentence, ETD aims to predict text spans and event types for event triggers based on a predefined set of event types, e.g., *"Attack"* and *"Transport"* (V. D. Lai et al., 2020). Similarly, EMR seeks to determine text spans and entity types (e.g., *"Person"*, *"Organization"*) for entity mentions in the sentence (T. H. Nguyen, Sil, Dinu, & Florian, 2016). Different from the first two tasks, EAE and RE involves predictions for a pair of objects at a time. Given an event trigger and an entity mention, EAE aims to predict the argument role (e.g, *"Victim"*) of the entity mention for the event trigger (Veyseh, Nguyen, & Nguyen, 2020a). An argument role can be *"Not-an-argument"* indicating that the entity mention is not an argument for the trigger. For RE (Veyseh, Dernoncourt, Dou, & Nguyen, 2020a; Veyseh, Dernoncourt, Thai, et al., 2020a), the task focuses on the classification of relation (e.g, *"Work for"*) for a given pair of entity mentions. There is also a special type *"No-relation"* to specify no relation between two entity mentions. As such, we call the union set C

of the predefined event types, entity types, argument roles, and relation types as the information types (excluding *"Not-an-argument"* and *"No-relation"*).

3.3.3 Model. To capture dependency among task instances for JointIE, an approach is to obtain all text spans for entity/event mention candidates along with their possible pairs to form the nodes for a dependency graph to improve representation learning. However, this approach will retain many text spans for non-entity/event mentions to introduce noise into the modeling. It will also entail a large dependency graph that can hinder the efficiency of the model. To this end, our model for JointIE first identifies text spans for entity mentions and event triggers. Afterwards, all possible pairs of event-entity and entity-entity mentions are considered to identify positive pairs for event arguments and relations respectively. The detected entity mentions, event triggers, event arguments, and relations are called task instances that should be classified to obtain corresponding information types in C. In our model, a dependency graph among the detected task instances will be learned to provide inputs for GCNs to compute dependency-enhanced representations for the task instances. Finally, the enhanced representations will be used to compute a joint distribution over labels for all the task instances to train our model. We will also employ Simulated Annealing to achieve the global optimum for label assignment of the task instances in the decoding phase.

3.3.3.1 Identifying event and entity mentions. Given an input sentence $\mathbf{w} = [w_1, \dots, w_N]$ with N words, we identify its event triggers and entity mentions by solving two corresponding sequence tagging problems for event and entity mentions. In particular, we use the BIO tagging schema to assign two labels to each word in $\mathbf{w}$ to mark the text spans of event triggers and entity mentions, i.e., {*"B-TRIGGER"*, *"I-TRIGGER"*, *"O"*} labels for event triggers, and {*"B-ENTITY"*, *"I-ENTITY"*, *"O"*} labels for entity mentions. The pre-trained transformer-based language model BERT (Devlin et al., 2019a) is first utilized to obtain the contextualized embeddings for the words in the sentence: $\mathbf{X} = \mathbf{x}_1, \ldots, \mathbf{x}_N = \text{BERT}([w_1, \ldots, w_N])$.

Next, the vector sequence $\mathbf{X}$ is sent to two different CRF layers (Chiu & Nichols, 2016; Lafferty et al., 2001) to compute two distributions for the tag sequences of $\mathbf{w}$ for event triggers and event mentions. The negative log-likelihoods L_t and L_e for golden trigger and entity tag sequences are then obtained to be included in the overall training loss. At test time, the Viterbi algorithm (Forney, 1973) is employed to determine the best tag sequences for event triggers and event mentions in $\mathbf{w}$.

Let V^t and V^e be the sets of text spans for event triggers and entity mentions respectively in $\mathbf{w}$ (i.e., golden spans in the training time and predicted spans in the test time). To prepare for the next components, we compute the representations vectors $\mathbf{z}_i^t$ and $\mathbf{z}_j^e$ for each event trigger/instance $t_i \in V^t$ and entity mention/instance $e_j \in V^e$ respectively by averaging over the contextualized embeddings of the words inside the spans.

3.3.3.2 Identifying event arguments and relations. Given the detected event triggers and entity mentions, we obtain a representation vector $\mathbf{z}_{ij}^a$ for each pair of event-entity mentions $a_{ij} = (t_i, e_j)$ (i.e., $t_i \in V^t$, $e_j \in V^e$), and a representation vector $\mathbf{z}_{ij}^r$ for each pair of entity-entity mentions $r_{ij} = (e_i, e_j)$ (i.e., $e_i, e_j \in V^e$) via:

$\mathbf{z}_{ij}^a = FFN_a^{down}(\text{concat}(\mathbf{z}_i^t, \mathbf{z}_j^e))$ and $\mathbf{z}_{ij}^r = FFN_r^{down}(\text{concat}(\mathbf{z}_i^e, \mathbf{z}_j^e))$.

Here, we use the feed-forward networks FFN_a^{down} and FFN_r^{down} to make sure that $\mathbf{z}_i^t$, $\mathbf{z}_j^e$, $\mathbf{z}_{ij}^a$, and $\mathbf{z}_{ij}^r$ have the same dimensionality. Next, the pair representation vectors $\mathbf{z}_{ij}^a$ and $\mathbf{z}_{ij}^r$ are sent into two different feed-forward networks followed by sigmoid activations to compute the possibilities for being positive examples for event arguments and relations of a_{ij} and r_{ij} respectively: $p_{ij}^a = \sigma(\text{FFN}^a(\mathbf{z}_{ij}^a))$, and $p_{ij}^r = \sigma(\text{FFN}^r(\mathbf{z}_{ij}^r))$. Here, $p_{ij}^a \in (0,1)$ is the probability for the entity mention e_j being an actual argument for the event trigger t_i while $p_{ij}^r \in (0,1)$ is the likelihood that there exists a relation of interest between the entity mentions e_i and e_j. At training time, we obtain the the negative log-likelihoods L_a and L_r for the golden event argument and relation identification to be included in the overall loss function for minimization. At test time, the event-entity pair a_{ij} and entity-entity pair r_{ij} are retained as positive examples for event arguments and relations if their likelihooods p_{ij}^a and p_{ij}^r are greater than 0.5.

For convenience, let V^a and V^r be the sets of positive event-entity pairs a_{ij} (called argument instances) and entity-entity pairs r_{ij} (called relation instances) respectively. Also, let $V = V^t \cup V^e \cup V^a \cup V^r$ be the set of all detected event, entity, argument, and relation instances. For each instance $v_i \in V$, we will use $\mathbf{v}_i$ for its corresponding instance representation (i.e., from $\mathbf{z}_i^t$, $\mathbf{z}_j^e$, $\mathbf{z}_{ij}^a$, or $\mathbf{z}_{ij}^r$).

3.3.3.3 Inducing Instance Dependency. Given the detected event, entity, argument, and relation instances in V, it remains to predict the information types in C for the instances to solve JointIE. While it is possible to directly employ the instance representations $\mathbf{v}_i$ for label prediction, our goal is to exploit instance dependency in IE to enhance the representation vector for one instance with the information from other instances to facilitate type prediction. In particular, using the instances v_i in V as the nodes in a dependency graph G, we aim to enrich instance representations by feeding them into a GCN model. As such, instead of assuming a heuristic manually-designed dependency graph among the instances as in previous work (M. V. Nguyen, Lai, & Nguyen, 2021; Zhang & Ji, 2021a), we propose to automatically learn the dependency graph G for the instances in V. To this end, our dependency graph G is a fully connected graph among the nodes in V where a weight $\alpha_{ij} \in (0,1)$ is learned for each edge to quantify the dependency between the instances v_i and v_j in V. In this work, we present two sources of information that can be used for determining the dependency between the task instances: (i) semantic and (ii) syntactic information.

Semantic Information: The semantic-based weight α_{ij}^{sem} for the edge between v_i and v_j quantifies their relatedness/dependency based on semantic information, i.e., via the representation vectors $\mathbf{v}_i$ and $\mathbf{v}_j$: $\alpha_{ij}^{sem} = FFN^{sem}(\text{concat}(\mathbf{v}_i, \mathbf{v}_i))$. Here, FFN^{sem} is a feed-forward network with the sigmoid function in the end.

Syntactic Information: The syntax-based weight α_{ij}^{syn} for the edge between v_i and v_j is computed in a similar way as α_{ij}^{sem}. In particular, for each word $w_k \in \mathbf{w}$, we retrieve the dependency relation d_k between w_k and its governor in the dependency tree of $\mathbf{w}$, which is generated by the Trankit's dependency parser (M. V. Nguyen, Lai, Veyseh, & Nguyen, 2021). We then obtain the embedding $\mathbf{m}_k$ of d_k for w_k by looking up the learnable dependency embedding matrix $\mathbf{M}$. Afterwards, the syntax-based representation vector u_i for the instance $v_i \in V$ is computed via: $u_i = \text{max-pool}_{w_k \in SPAN_{v_i}}(\mathbf{m}_k)$. Here, $SPAN_{v_i}$ involves the words in the corresponding text span of v_i in $\mathbf{w}$ if v_i is an event trigger or entity mention instance. Otherwise, $SPAN_{v_i}$ contains the words inside the text spans of the involving event triggers and entity mentions in the pair for v_i. As such, we compute the syntax-based dependency weight α_{ij}^{syn} for v_i and v_j via:

$\alpha_{ij}^{syn} = FFN^{syn}(\text{concat}(\mathbf{u}_i, \mathbf{u}_i))$ where FFN^{syn} is also a feed-forward network with the sigmoid function in the end. Finally, we combine the semantic- and syntax-based weights to obtain the overall dependency weight α_{ij} for v_i and v_j in V: $\alpha_{ij} = (\alpha_{ij}^{sem} + \alpha_{ij}^{syn})/2$.

3.3.3.4 Enhancing Representations with GCNs. To enhance the representation vectors for the instances $v_i \in V$, a GCN model with K layers is applied over the induced dependency graph G to compute richer representations for the instances:

$$\mathbf{h}_i^k = \text{ReLU}(\frac{\sum_{v_j \in V} \alpha_{ij} \mathbf{W}^k \mathbf{h}_j^{k-1} + \mathbf{b}^k}{\sum_{v_j \in V} \alpha_{ij}}), 1 \leq k \leq K \tag{3.3}$$

Here, h_i^k is the representation for the instance v_i at the k-th layer of the GCN ($\mathbf{h}_i^0 \equiv \mathbf{v}_i$), and $\mathbf{W}^k, \mathbf{b}^k$ are trainable weight and bias for the layer.

In this way, representation information from all the other instances v_j $(j \neq i)$ will be incorporated into the enhanced representation vector for v_i according to their learned dependency weights. Finally, the last layer's representation $\mathbf{h}_i^K \equiv \mathbf{h}_i$ (we omit K for simplicity) is used to compute the score vector $\mathbf{s}_i \in \mathbb{R}^{|C|}$ for v_i, where $\mathbf{s}_i[c]$ measure the possibility for v_i to have the c-th label in the label set C: $\mathbf{s}_i = FFN^{score}(\mathbf{h}_i)$ (FFN^{score} is a scoring feed-forward network). The score vectors $\mathbf{s}_i$ will later be used for modeling the joint distribution of the labels for all the instances in V.

3.3.3.5 Computing Joint Distribution of Labels. Let Y be the set of labels y_i for the instances v_i in V. To infer the labels for the instances in V, we need to estimate the joint distribution $P(Y|\mathbf{w}, V)$. In previous work (Y. Lin et al., 2020a; M. V. Nguyen, Lai, & Nguyen, 2021; Wadden et al., 2019a; Zhang & Ji, 2021a), JointIE methods mostly focus on learning representations for the task instances to compute a label distribution for each instance v_i for prediction:

$P(y_i|\mathbf{w}, V) \coloneqq \text{softmax}(\mathbf{s}_i)$. This practice essentially implies the following factorization for $P(Y|\mathbf{w}, V)$: $P(Y|\mathbf{w}, V) = \prod_{y_i \in Y} P(y_i|\mathbf{w}, V)$. As a result, this factorization assumes the independence of the instance labels, thus unable to fully capture beneficial label dependency for IE tasks.

To address this issue, we directly estimate the joint distribution $P(Y|\mathbf{w}, V)$ so that the dependency between instance labels can be facilitated to improve prediction performance. To this end, we formulate the joint distribution $P(Y|\mathbf{w}, V)$ with Conditional Random Fields (Lafferty et al., 2001):

$$P(Y|\mathbf{w}, V) = \frac{1}{Z(V)} \prod_{(v_i, v_j)} \psi_{ij}(y_i, y_j, V) \tag{3.4}$$

where $\psi_{ij}(y_i, y_j, V)$ is a positive potential function defined on the edge (v_i, v_j) of the dependency graph G, and $Z(V) = \sum_{Y' \in C_V} \prod_{(v_i, v_j)} \psi_{ij}(y'_i, y'_j, V)$ is the normalization term to make sure that $P(Y|\mathbf{w}, V)$ is a valid probability distribution (C_V is the set of all possible label assignments Y for the instances in V). Considering the instance information, the instance dependency, and the label dependency, we propose the potential function as:

$$\psi_{ij}(y_i, y_j, V) \coloneqq \exp(\mathbf{s}_i[y_i] + \mathbf{s}_j[y_j] + \alpha_{ij} \pi_{y_i \leftrightarrow y_j}) \tag{3.5}$$

where $\mathbf{s}_i[y_i]$ is the local score for instance v_i being assigned with the label y_i, α_{ij} is the induced dependency weight for the edge (v_i, v_j) in G, and $\pi_{y_i \leftrightarrow y_j}$ is a learnable transition score indicating the dependency between the labels y_i and y_j. With this formulation, we can derive the joint distribution $P(Y|\mathbf{w}, V)$:

$$P(Y|\mathbf{w}, V) = \frac{\exp(s(Y))}{\sum_{Y' \in C_V} \exp(s(Y'))} \tag{3.6}$$

where:

$$s(Y) = \gamma \sum_{v_i \in V} \mathbf{s}_i[y_i] + \sum_{(v_i, v_j)} \alpha_{ij} \pi_{y_i \leftrightarrow y_j} \tag{3.7}$$

is the global score for the label assignment/configuration Y of the instances. γ is a hyper-parameter to balance the local and transition scores.

To train the model, we need to maximize the joint likelihood in Equation (3.6) for the golden label configuration Y^*. However, this requires the computation of the normalization term $\sum_{Y' \in C_V} \exp(s(Y'))$, which is intractable. To overcome this issue, we employ Noise Contrastive Estimation (NCE) (Gutmann & Hyvärinen, 2012; Mikolov et al., 2013). NCE converts the maximization problem into the nonlinear logistic regression that discriminates between the golden label configurations and the noise label configurations. In particular, the maximization of $P(Y^*|\mathbf{w}, V)$ is done with NCE via minimizing the contrastive loss:

$$L_{NC} = -\text{log}\sigma(s(Y^*)) - \sum_{n=1}^{N_{noi}} \mathbb{E}_{Y'_n \sim P_{noi}} \left[\text{log}\sigma(-s(Y'_n))\right] \tag{3.8}$$

where σ is the sigmoid function and N_{noi} is the number of noise configurations Y'_n drawn from P_{noi}, assumed to be a uniform distribution. Intuitively, the minimization of L_{NC} increases the global score $s(Y^*)$ for the true label configuration Y^* while decreasing the global scores $s(Y')$ for the noise label configurations Y' to appropriately train the model. To the end, the overall loss function to train our model is: $L = L_t + L_e + L_a + L_r + L_{NC}$.

3.3.3.6 Joint Decoding via Simulated Annealing. At inference time, we need to search for the configuration $\hat{Y}$ that has the highest global score $s(\hat{Y})$ in C_V: $\hat{Y} = \text{argmax}_{Y' \in C_V} s(Y')$. A brute-force search for $\hat{Y}$ cannot be done as the search space C_V is exponentially large ($|C_V| = |C|^{|V|}$). Previous work has made several attempts to deal with this issue. (Wadden et al., 2019a) and (M. V. Nguyen, Lai, & Nguyen, 2021) simply perform greedy decoding for each instance label independently, thus unable to exploit the label dependency. (Y. Lin et al., 2020a) and (Zhang & Ji, 2021a) resort to beam search that step by step constructs a

Algorithm 1: Simulated Annealing Search

```
   Input: $\hat{Y}_0$ where $\hat{y}_{i,0} = \text{argmax}_{c \in C}\mathbf{s}_i[c]$.
1  $\hat{Y}_{cur} \leftarrow \hat{Y}_0$; $n \leftarrow 1$;
2  while $n \leq N_{iter}$ do
3      $t \leftarrow T/n$;
4      if $t < \epsilon$ then
5          return $\hat{Y}_{cur}$;
6      else
7          $\hat{Y}_{new} = random\_successor(\hat{Y}_{cur})$;
8          $\delta_n = s(\hat{Y}_{new}) - s(\hat{Y}_{cur})$;
9          if $\delta_n > 0$ then
10             $\hat{Y}_{cur} \leftarrow \hat{Y}_{new}$;
11         else
12             $\hat{Y}_{cur} \leftarrow \hat{Y}_{new}$ with $p = \exp(\frac{\delta_n}{t})$ ;
13         end
14     end
15     $n \leftarrow n + 1$;
16 end
17 return $\hat{Y}_{cur}$.
```

complete decoding assignment Y for the instances in V by expanding an initially empty assignment. Each step corresponds to an instance in V where only top candidate labels for the instance are considered for assignment expansion and only top partial assignments produced so far are kept for the next step. Unfortunately, the selection of top candidate labels for expansion at each step is based only on the local scores $\mathbf{s}_i$, which might discard the candidates that can eventually provide greater global scores. To overcome this issue, we propose to apply Simulated Annealing (SA) (Kirkpatrick et al., 1983) to search for the optimal assignment $\hat{Y}$ for V. SA is a probabilistic algorithm that is able to approximately find the global optimum of a function (Kirkpatrick et al., 1983; Van Laarhoven & Aarts, 1987). Algorithm 1 presents our implementation for SA to find $\hat{Y}$.

The input for the algorithm is the initial configuration $\hat{Y}_{cur} = \hat{Y}_0 = \{\hat{y}_{i,0}\}$, which contains the greedily predicted labels for each instance: $\hat{y}_{i,0} = \text{argmax}_{c \in C}\mathbf{s}_i[c]$.

Datasets	**Split**	**#sents**	**#ents**	**#rels**	**#events**
ACE05-R	Train	10,051	26,473	4,788	-
	Dev	2,424	6,362	1,131	-
	Test	2,050	5,476	1,151	-
ACE05-E	Train	17,172	29,006	4,664	4,202
	Dev	923	2,451	560	450
	Test	832	3,017	636	403
ACE05-E+	Train	19,240	47,525	7,152	4,419
	Dev	902	3,422	728	468
	Test	676	3,673	802	424
ERE-EN	Train	14,219	38,864	5,045	6,419
	Dev	1,162	3,320	424	552
	Test	1,129	3,291	477	559
ERE-ES	Train	7,067	11,839	1,698	3,272
	Dev	556	886	120	210
	Test	546	811	108	269

Table 16. Data statistics. **#sents**, **#ent**, **#rels**, and **#events** indicate the number of sentences, entity mentions, relations, and events respectively.

The algorithm then runs over N_{iter} iterations to improve the global score $s(\hat{Y}_{cur})$ for the current label configuration $\hat{Y}_{cur}$. This is done via updating the current configuration to a successor configuration $\hat{Y}_{new}$ that gives a higher global score (i.e., $\delta_n > 0$). A successor configuration is obtained via the function $random\_successor()$ by randomly changing some label $\hat{y}_i \in \hat{Y}_{cur}$. Different from beam search decoding with partial assignments, each searching step in SA examines a complete label assignment for the instances in V to provide complete information to measure the global scores/quality of the assignments. Importantly, SA sometimes allows the current configuration to transition to a successor configuration with a lower global score (i.e., $\delta_n \leq 0$) with an acceptance probability of $p = \exp(\frac{\delta_n}{t})$. Here, t is the temperature of the algorithm, gradually decreased via $t \leftarrow T/n$ (T is a hyper-parameter). This exploration property enables SA to escape from local optimum configurations, thus increasing the chance to find the globally optimal configuration $\hat{Y}$.

PLMs	**Model**	**ACE05-R**		**ACE05-E**			**ACE05-E+**				**ERE-EN**				**ERE-ES**			
		Ent	Rel	Ent	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg
T5	Text2event	-	-	-	71.9	53.8	-	-	71.8	54.4	-	-	59.4	48.3	-	-	-	-
BART	DEGREE	-	-	-	72.2	56.0	-	-	71.7	58.0	-	-	56.6	51.1	-	-	-	-
BERT	OneIE	88.6	63.4	90.2	74.7	56.8	89.6	58.6	72.8	54.8	87.0	53.2	57.0	46.5	81.3	48.1	56.8	40.3
	AMRIE*	88.7	67.2	90.8	75.3	58.2	90.4	62.9	72.8	56.3	86.9	55.5	58.3	44.2	-	-	-	-
	FourIE	88.9	68.9	**91.3**	75.4	58.0	**91.1**	63.6	73.3	57.5	**87.4**	56.1	57.9	48.6	**82.2**	57.9	57.1	42.3
	GraphIE	**88.9**	**69.5**	90.6	**75.7**	**58.8**	91.0	**65.4**	**74.8**	**59.9**	87.2	**57.8**	**61.4**	**52.2**	81.4	**58.9**	**61.3**	**45.7**
RoBERTa	OneIE*	89.0	65.2	90.2	74.7	55.6	90.8	60.4	72.5	56.3	86.3	52.8	57.1	47.1	83.7	57.5	58.3	42.5
	AMRIE	89.2*	66.8*	**92.1**	75.0	58.6	91.0*	62.8*	72.7*	57.7*	87.9	55.2	61.4	45.0	-	-	-	-
	FourIE*	89.1	67.5	91.6	74.9	58.7	91.1	63.1	72.8	58.3	**88.0**	56.2	61.5	49.1	83.9	61.0	62.3	44.2
	GraphIE	**89.3**	**68.5**	91.4	**75.1**	**59.4**	**91.6**	**66.0**	**73.3**	**60.2**	87.7	**57.0**	**62.0**	**54.7**	**84.3**	**62.3**	**65.7**	**46.9**

Table 17. Model performance on the test data of 5 datasets. *"Ent"*, *"Rel"*, *"Trg"*, and *"Arg"* are the F1 scores for identification and classification of entity mentions, event triggers, relations, and event arguments respectively. * indicates results that are not reported in the original papers but produced by their official code. Underlined numbers designate the tasks where *GraphIE* is significantly better (p ¡ 0.01) than the baselines.

3.3.4 Experiments.

Datasets: Following previous work (I. Hsu et al., 2021; Y. Lin et al., 2020a; Lu et al., 2021a; M. V. Nguyen, Lai, & Nguyen, 2021; Wadden et al., 2019a; Zhang & Ji, 2021a), we conduct experiments on 5 different datasets created by the 2005 Automatic Content Extraction (ACE05) (Walker et al., 2006) and Entity Relation Event (ERE) (Song et al., 2015) programs. The three ACE05 datasets feature **ACE05-R**, **ACE05-E**, and **ACE-E+**, all in English, involving 33 event types, 7 entity types, 6 relation types, and 22 argument roles. The two ERE datasets are **ERE-EN** (English portion) and **ERE-ES** (Spanish portion), introducing 38 event types, 7 entity types, 5 relation types, and 20 argument roles. We use the same data processing and train/dev/test splits as the prior work for a fair comparison. Detailed statistics for the datasets are shown in Table 16.

Baselines: We compare our method, called *GraphIE*, with the following baselines for JointIE:

Generative baselines: *Text2event* (Lu et al., 2021a) and *DEGREE* (I. Hsu et al., 2021). The generative baselines perform ETD and EAE via formulating the tasks as text generation. The models receive an input sentence and generate an output text containing text spans and labels for event triggers and event arguments, structured in a way that a post-processing step can be used to extract ETD and EAE predictions for the models.

Classification baselines: *OneIE* (Y. Lin et al., 2020a), *AMRIE* (Zhang & Ji, 2021a), and *FourIE* (M. V. Nguyen, Lai, & Nguyen, 2021). The classification baselines represent the instances for ETD, EMR, EAE, and RE via a shared encoder and perform classification for the instances based on task-specific label distributions. *AMRIE* and *FourIE* employ a heuristic dependency graph among task instances to improve representation learning. Dependency between instance labels is exploited in *OneIE* and *AMRIE* via a beam search decoding with manually-designed global features, and in *FourIE* via global type dependency regularization. *FourIE* and *AMRIE* are the current state-of-the-art models for JointIE.

Hyper-parameters: Prior work for JointIE employs two different versions of pre-trained language models (PLM), i.e., *BERT* (Devlin et al., 2019a; Y. Lin et al., 2020a; M. V. Nguyen, Lai, & Nguyen, 2021) and *RoBERTa* (Y. Liu et al., 2019; Zhang & Ji, 2021a), which might cause incompatible compassion. To this end, we explore both BERT and RoBERTa to obtain the word representations $\mathbf{x}_i$ for *GraphIE* for a fair comparison. For the Spanish ERE-ES dataset, following prior work (Y. Lin et al., 2020a; M. V. Nguyen, Lai, & Nguyen, 2021), we utilize the multilingual versions of BERT and RoBERTa. For each PLM, we fine-tune the hyper-parameter for *GraphIE* on the development data.

In particular, the best values for the hyper-parameters of the proposed model are reported as follows. We employ the learning rate of $1e-5$ for the models with the BERT-based PLM (i.e., using *bert-large-cased* and *bert-multilingual-cased*) and the learning rate of $5e-6$ for the RoBERTa-based PLM (i.e., using *roberta-large* and *xlm-roberta-large*). For other hyper-parameters, our tuning process results in the same values for BERT-based and RoBERT-based models: Adam (Kingma & Ba, 2014) for the optimizer, batch size of 10, 100 for the size of the dependency relation embeddings, 400 for the size of the hidden vector for the feed-forward networks, 200 for the hidden vector size in the GCN model, 2 for the number of layers for the feed-forward networks and GCN model, $\gamma = 1$ for the trade-off hyper-parameter for the global score, $N_{noi} = 5$ for the number of noise examples for the contrastive loss (we re-sample the noise examples every epoch), $T = 5$ for the initial temperature, $N_{iter} = 50$ for the number of iterations of Simulated Annealing (SA), and $\epsilon = 0.1$ for the temperature threshold for the SA decoding.

Comparison with Baselines: We compare the proposed model *GraphIE* with the baselines on test data of the 5 datasets in Table 17. As can be seen, the generative baselines perform worse than the classification models on most of the settings. This might be due the implicit modeling of the label distributions and the assumption of a decoding order for task instances that limit the interactions of instance labels. Comparing *OneIE*, *FourIE* and *AMRIE*, it is clear that the exploitation of instance and label dependency in the training phase in *FourIE* can lead to better performance for JointIE than using such dependency in the decoding phase as done by *OneIE* and *AMRIE* over most tasks and PLMs. Most importantly, the proposed *GraphIE* significantly outperforms all the baselines across a majority of settings for tasks, datasets and PLMs, thus demonstrating the benefits of induced dependency graph, joint label distribution estimation, and simulated annealing for decoding in our method.

Model *(all use Roberta)*	**ACE05-E+**			
	Ent	Rel	Trg	Arg
GraphIE	**89.8**	**67.2**	**72.6**	**66.3**
- *induced dep*	89.3	65.8	71.3	65.0
- *semantic-based dep*	89.0	66.4	71.6	65.9
- *syntactic-based dep*	89.4	66.3	72.0	65.4
- *induced dep + heuristic dep*	89.3	66.2	71.7	65.5
- *GCN*	89.4	65.6	70.9	64.6

Table 18. Performance (F1) on the ACE05-E+ development data.

Ablation Study: To understand the contributions of each proposed component to *GraphIE*, we conduct ablation experiments where we remove each component from the full model and evaluate the performance of the remaining models.

The first three ablated models in Table 18 are *"- induced dep"*, *"- semantic dep"*, and *"- syntactic dep"*, formed by excluding the dependency weight induction of α_{ij} (i.e., setting $\alpha_{ij} = 1$), the semantic-based dependency α_{ij}^{sem}, and the syntactic-based dependency α_{ij}^{syn} (respectively) from the model computation. In each case, the performance of *GraphIE* decreases significantly; the removal of both semantic- and syntactic-based dependency in *"- induced dep"* leads to the largest performance drop. This shows that the semantic and syntactic weighting captures complementary information for instance dependency induction that is useful for our model. The next ablated model *"- induced dep + heuristic dep"* is obtained by replacing the induced dependency graph represented by α_{ij} with the heuristic dependency graph for instances from the best baseline *FourIE*. The decrease in the performance of this model suggests that the induced dependency graph is better than the heuristic graph for JointIE. The final ablated model *"- GCN"* in Table 18 eliminates the GCN component from our full model. The result shows that GCN is beneficial to exploit the induced dependency graph to improve representation learning.

Model *(all use Roberta)*	**ACE05-E+**			
	Ent	Rel	Trg	Arg
GraphIE	**89.8**	**67.2**	**72.6**	**66.3**
- joint distribution	89.3	65.5	70.9	64.5
- SA + greedy	89.2	65.9	71.2	65.2
- SA + beam	89.5	66.0	71.5	65.4
- SA + hill climbing	89.5	66.8	71.7	65.3
OneIE	88.7	64.2	69.5	63.2
- beam + SA	88.1	63.9	69.1	62.7
AMRIE	89.4	65.4	71.2	64.4
- beam + SA	88.8	65.1	70.5	64.1

Table 19. Performance (F1) on the ACE05-E+ development data.

In Table 19, we first eliminate the computation of the joint label distribution $P(Y|\mathbf{w}, V)$ from *GraphIE*. As such, the *"- joint distribution"* model employs the local label distributions $P(y_i|\mathbf{w}, V)$ to train models and infer labels (with greedy decoding). Due to the significantly worse performance of *"- joint distribution"*, it is clear that directly estimating the joint label distribution is helpful for JointIE. To evaluate the benefit of the proposed SA, we replace it with other decoding algorithms for *GraphIE*, including greedy search, beam search and hill climbing. The beam search is implemented with our global score function $s(Y)$ and follows those in (Y. Lin et al., 2020a; Zhang & Ji, 2021a) while hill climbing is implemented by removing the configuration exploration in lines 11-12 of Algorithm 1. As reported in Table 19, SA performs much better than other decoding algorithms for *GraphIE*, thus demonstrating SA's ability to find globally optimal labels. In addition, we also attempt to replace the beam search decoding in *OneIE* and *AMRIE* with SA, which indeed leads to worse performance for such models as shown in the last four rows of Table 19. We attribute this to the learning of the global scores for configurations in *OneIE* and *AMRIE* that involves a limited set of predefined global features. Such features do not exist for many possible assignments Y for V, thus causing poor global score computation and hindering the configuration ranking critically required by SA.

Label pair	**Transition score**
(Argument:Origin, Argument:Place)	10.02
(Event:Transport, Relation:Physical)	4.33
(Relation:Org-Aff, Relation:Part-Whole)	3.58
(Event:Execute, Event:Sentence)	2.58
(Event:Die, Event:Be-Born)	-2.34
(Event:Attack, Argument:Origin)	-87.07
(Relation:Per-Soc, Entity:Facility)	-93.93
(Transport, Attacker)	-99.91

Table 20. Transition scores for some label pairs learned by our model on ACE05-E+.

Analysis: To further understand the advantages of *GraphIE* over baseline models, we manually analyze the instances on the ACE05-E+ development data where *GraphIE* can make correct predictions, but the best baseline model *FourIE* fails. Figure 15 presents some instances along with their edges and weights in the dependency graphs. The most important insight from our analysis is that *GraphIE* is able to connect an instance (e.g., *blew*) with other supporting instances (e.g., *suicide*) in the dependency graph to provide vital information to facilitate correct prediction. Such supporting instances do not share any event trigger or entity mention with the current instance that cannot establish links in *FourIE* and lead to failure predictions.

Finally, Table 20 shows the transition scores $\pi_{y_i \leftrightarrow y_j}$ learned by *GraphIE* for some label pairs in ACE05-E+. The table show that our model is able to learn high scores for correlated label pairs (e.g., the *Execute* and *Sentence* event types) and very low scores for uncorrelated label pairs (e.g., an argument for a *Transport* event cannot play the role *Attacker*).

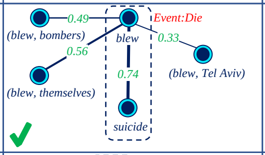
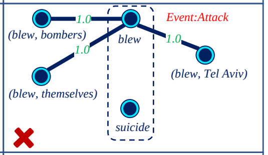
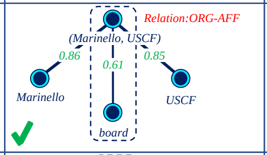
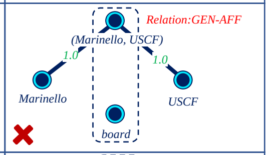
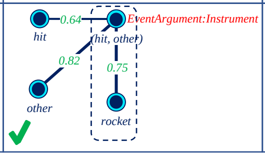
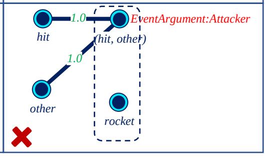
Example	GraphIE	FourIE
In the January attack, two Palestinian <u>suicide</u> bombers blew themselves up in central Tel Aviv, killing 23 other people. ***Explanation**: "blew" is correctly predicted by GraphIE as a "Die" event trigger while FourIE incorrectly predicted it as an "Attack" event trigger.*	 0.49 (blew, bombers) blew Event:Die 0.33 0.56 0.74 (blew, Tel Aviv) (blew, themselves) suicide ✓	 1.0 (blew, bombers) blew Event:Attack 1.0 1.0 (blew, Tel Aviv) (blew, themselves) suicide ✗
We pretty much know that Marinello, while on the <u>board</u>, has arranged to get future money from the USCF. ***Explanation**: The relation between "Marinello" and "USCF" is correctly predicted by GraphIE as a "ORG-AFF" relation while FourIE incorrectly predicted it as a "GEN-AFF" relation.*	 Relation:ORG-AFF (Marinello, USCF) 0.86 0.61 0.85 Marinello USCF board ✓	 Relation:GEN-AFF (Marinello, USCF) 1.0 1.0 Marinello USCF board ✗
A second <u>rocket</u> landed in farmlands and the other hit a house inside the refugee camp, … ***Explanation**: "other" is correctly predicted by GraphIE as an "Instrument" for the event trigger "hit" while FourIE incorrectly predicted it as an "Attacker" for the event trigger "hit".*	 0.64 hit EventArgument:Instrument (hit, other) 0.82 0.75 other rocket ✓	 1.0 hit EventArgument:Attacker (hit, other) 1.0 other rocket ✗

Figure 15. Instances along with their dependency subgraphs in ACE05-E+. Supporting instances are underlined.

3.3.5 Related Work.

Capturing dependency between IE tasks has been a main focus of previous work on Joint IE. Early work employed feature engineering methods (Q. Li et al., 2013a; Roth & Yih, 2004a; B. Yang & Mitchell, 2016a; Yu & Lam, 2010a). Later work applied deep learning via shared parameters to facilitate joint modeling for IE, however, for only two or three tasks (Bekoulis et al., 2018a; Luan et al., 2019a; T. H. Nguyen, Cho, & Grishman, 2016a; T. M. Nguyen & Nguyen, 2019a; Zhang et al., 2019; Zheng et al., 2017a). Recently, the four IE tasks have been solved jointly (Y. Lin et al., 2020a; Lu et al., 2021a; M. V. Nguyen, Lai, & Nguyen, 2021; Paolini et al., 2021; Wadden et al., 2019a; Zhang & Ji, 2021a). However, such recent works only employ heuristics to manually design dependency graphs for instances. Mean-field factorization of the joint label distribution for JointIE instances is dominant in prior work.

Our work is also related to prior work that uses CRFs (Chiu & Nichols, 2016; Lafferty et al., 2001) to estimate joint distribution of instance labels. Sequence labeling is a typical problem that has been solved by CRFs, including part of speech tagging and named entity recognition (Chiu & Nichols, 2016; Ekbal, Haque, & Bandyopadhyay, 2007; Lafferty et al., 2001; Shishtla, Gali, Pingali, & Varma, 2008; Sobhana, Mitra, & Ghosh, 2010; K. Xu, Zhou, Hao, & Liu, 2017; Zea, Luna, Thorne, & Glavaš, 2016). However, these prior work only employ CRFs for simple graph structures (i.e., linear chains). A few prior work has considered CRFs for more complicated graph structures (Gao, Pei, & Huang, 2019; Qu, Bengio, & Tang, 2019; X. Sun, Lin, Shen, & Hu, 2017; H. Yuan & Ji, 2020); however, none of such works has applied CRFs for JointIE as we do.

3.3.6 Summary. We propose a novel model for jointly solving four IE tasks (EMR, ETD, EAE, and RE). Our proposed model learns a dependency graph among the instances of the tasks via a novel edge weighting mechanism. We also estimate the joint distribution among instance labels to fully enable interactions between instance labels for improved performance. The experimental results show that our model achieves best performance for multiple JointIE tasks across 5 datasets and 2 languages.

CHAPTER IV

LEARNING METHODS FOR IE IN LOW-RESOURCE LANGUAGES

This chapter contains materials from the published papers *"Minh Nguyen, Tuan Ngo Nguyen, Bonan Min, and Thien Huu Nguyen.* ***'Crosslingual Transfer Learning for Relation and Event Extraction via Word Category and Class Alignments'*** *In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, 2021"* (M. V. Nguyen, Nguyen, et al., 2021) and *"Minh Nguyen, Nghia Trung Ngo, Bonan Min, and Thien Huu Nguyen.* ***'FAMIE: A Fast Active Learning Framework for Multilingual Information Extraction'*** *In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations, 2022"* (M. V. Nguyen, Ngo, et al., 2022). Minh was responsible for the method design, experiments, evaluation and writing as the first author. Tuan, Nghia, Bonan, and Thien provided meaningful discussions and analysis. Thien contributed to the method design and editorial revisions for the paper submissions. The papers were revised to comply with the dissertation format and purposes.

The third research direction (RD3) addresses the challenge of non-existent or limited training data in target languages for multilingual IE. This chapter focuses on two scenarios: (1) when training data is unavailable in the target languages, and (2) when limited training data is available in the target languages. For the first scenario, we present our novel learning method called CCCAR for class- and word category-based crosslingual alignment of representations. CCCAR ensures similar representations of the same concepts across source and target languages, improving the cross-lingual transferability of the model. For the second scenario, we introduce

FAMIE, a novel active learning framework that employs a small proxy network for fast data selection and annotation, maximizing the performance of IE models in the target languages. Extensive experiments demonstrate the effectiveness of CCCAR and FAMIE in enhancing multilingual IE in low-resource settings.

4.1 CCCAR

4.1.1 Introduction.

Relation and Event Extraction (REE) are important tasks of Information Extraction (IE), whose goal is to extract structured information from unstructured text (Walker et al., 2006). Due to their complexity, annotations for REE tasks are costly and only available in a few languages. Thus, there have been growing interests on crosslingual learning for REE in which a model is trained on a language, i.e., source language, and applied to another language, i.e., target language, where the annotations are not available. Recent approaches for crosslingual REE have mainly employed multilingual word embeddings, e.g., MUSE, (Joulin, Bojanowski, Mikolov, Jégou, & Grave, 2018; J. Liu et al., 2019a; Ni & Florian, 2019; Subburathinam et al., 2019) or multilingual pre-trained language models, e.g., multilingual BERT, (Ahmad, Peng, & Chang, 2021; Devlin et al., 2019a; M'hamdi et al., 2019; M. V. Nguyen & Nguyen, 2021b) to learn crosslingual representation vectors for REE.

However, previous work on crosslingual REE suffers from the monolingual bias issue due to the monolingual training of models on only the source language data, leading to non-optimal crosslingual performance. A solution for this issue can resort to language adversarial training (X. Chen et al., 2019; He, Yan, & Xu, 2020; Huang et al., 2019; Keung, Lu, & Bhardwaj, 2019; Lange, Iurshina, Adel, & Strötgen, 2020a) where unlabeled data in the target language is used to aid crosslingual representations via fooling a language discriminator. The underlying

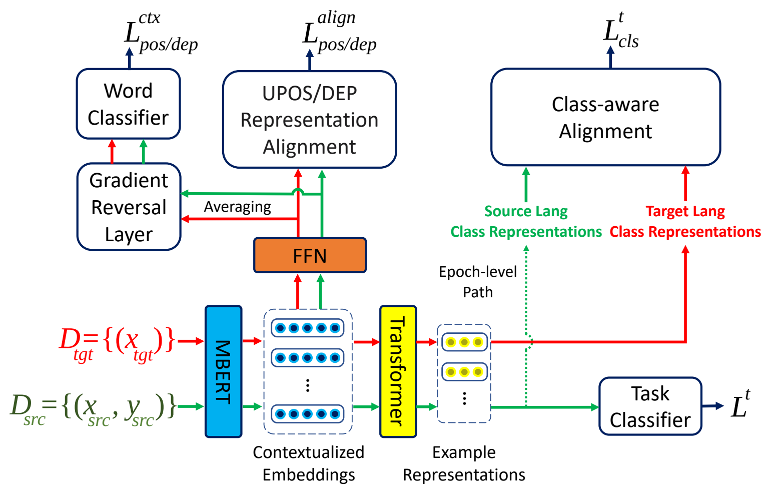


Figure 16. Overall architecture of the proposed models for RE, EAE. For ED, example representations are the contextualized embeddings.

principle for this approach is to encourage the closeness of representation vectors for sentences in the source and target languages (i.e., aligning representation vectors). However, a critical drawback of language adversarial training is the failure to condition on classes/types of examples in the alignment process. As such, a target language example of a class could be incorrectly aligned to a source language example of a different class in REE, causing confusion and hindering the performance of the models. The middle sub-figure in Figure 17 demonstrates the class misalignment of representation vectors in crosslingual REE.

To this end, we propose a crosslingual alignment method that explicitly conditions on class information of REE tasks to enhance representation alignment and learning. Our major intuition is that the semantics of the classes in REE tasks (e.g., the event type of *Attack* in event extraction) are generally invariant across languages that can be leveraged as anchors to bridge representation vectors for examples in different languages. As such, we can obtain two semantic representation vectors for each class in an REE task based on representation vectors of examples in either source or target language. Afterward, the representation vectors of the same class can be regulated to match each other, serving as a mechanism for class-aware crosslingual alignment of representation vectors for source and target examples. To implement this idea, we use multilingual BERT (mBERT) to obtain same-space representations for examples in both source and target languages to facilitate the alignment process. Afterward, the source-language representation vector for a class is computed via representation vectors of source-language examples that belong to the corresponding class. For the target language, as class information is not provided, we seek to compute target-language representation vector for a class by aggregating representation vectors for unlabeled examples, weighted on an estimation of the probabilities for the examples to exhibit the class.

In addition to class semantics, we propose to further exploit universal parts of speech and dependency relations in parsing trees (i.e., word categories) to improve the cross-lingual alignment for representation vectors in REE. As such universal word categories have been consistently annotated for more than 100 languages (Zeman et al., 2020) and can be generated with high accuracy via existing toolkits, e.g., the transformer-based toolkit Trankit for multilingual NLP (M. V. Nguyen, Lai, Veyseh, & Nguyen, 2021; Qi, Zhang, Zhang, Bolton, & Manning, 2020a; Straka, 2018a), we expect this information to provide helpful anchor knowledge for cross-lingual representation learning. To this end, similar to the class-aware alignment, we propose to align representation vectors of the same universal word categories that are computed using contextualized representations of examples in the source and target languages to further improve the language-independence of representation vectors for REE.

A potential issue with the computation of word category representations via contextualized representations of examples is the preservation of context word information in representations for word categories that might introduce noise and hinder the representation alignment. To address this issue, we propose an adversarial training model that seeks to explicitly filter context information from word category representations. This is achieved by using Gradient Reversal Layer (Ganin & Lempitsky, 2015) to prevent word category representations from being able to recognize the context words in the original examples. We expect that this filtering mechanism can improve the word category pureness of the representations, thus providing appropriate inputs for the alignment process for improved representation learning.

We conduct extensive experiments with different crosslingual settings on English, Chinese, and Arabic for three REE tasks, i.e., Relation Extraction, Event Detection, and Event Argument Extraction. The results demonstrate the benefits of the proposed method that significantly advances the state-of-the-art performance in these settings.

4.1.2 Problem Statement. We study cross-lingual transfer learning for three REE tasks as defined in the ACE 2005 dataset (Walker et al., 2006), i.e., Relation Extraction (RE), Event Detection (ED), and Event Argument Extraction (EAE). Given two entity mentions in an input sentence, the goal of RE is to determine the semantic relationship between the mentions according to predefined relation types/classes (e.g., *Employment*). For ED, its purpose is to identify event triggers, which can be verbs/normalization with one or multiple words, that express occurrences of events of predefined types (e.g., *Attack*). Finally, given an event trigger and an entity mention, EAE aims to predict the role (e.g., *Victim*) that the entity mention plays in the corresponding event. Note that, we have a special type *None* to indicate non-relation, non-trigger, or non-argument for RE, ED, and EAE respectively.

For further discussion, let $D_{src} = \{(x_{src}, y_{src})\}$ ($|D_{src}| = N_{src}$) be the labeled training set in the source language. As such, for ED, x_{src} is an input sentence and y_{src} serves as the golden sequence tag (using BIO) for the words in x_{src}. For RE and EAE, x_{src} involves an input sentence along with indexes of the given trigger word and entity mentions while y_{src} represents the golden relation type or argument role for the input. We also assume access to an unlabeled dataset $D_{tgt} = \{(x_{tgt})\}$ ($|D_{tgt}| = N_{tgt}$) in the target language where x_{tgt} consists of similar information as x_{src} for the corresponding task.

4.1.3 Baseline Methods. To prepare for our cross-lingual representation alignment techniques for REE, we first describe the baseline models explored in this work.

4.1.3.1 Using Source Language Data Only. In this section, we present two baselines that train models based only on labeled data in the source language. These baselines are the current state-of-the-art (SOTA) models for crosslingual transfer learning for ED, RE, and EAE on the ACE 2005 dataset (Walker et al., 2006).

BERTCRF (M'hamdi et al., 2019): This is the current SOTA model for crosslingual ED. Given an input sentence $\mathbf{w} = [w_1, w_2, \ldots, w_n]$ with n words (in x_{src}), the model first sends $\mathbf{w}$ to the mBERT encoder to obtain a sequence of contextualized representations $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n]$ where $\mathbf{z}_k$ is the representation for each $w_k \in \mathbf{w}$, computed as the average of its word-piece representations returned by the last layer of mBERT. The ED task is then done by performing sequence labeling over the words in $\mathbf{w}$ where each word is assigned with a BIO tag to capture boundaries and event types of event triggers in $\mathbf{w}$. In particular, the final representation vector for trigger prediction $\mathbf{r}^{ED}_{src,k}$ is directly formed from the word representation $\mathbf{z}_k$ (i.e., $\mathbf{r}^{ED}_{src,k} = \mathbf{z}_k$). Afterward, this prediction representation is fed into a feed-forward network FFN^{ED} to obtain a score vector that exhibits the likelihoods for w_k to receive possible BIO tags for the predefined event types: $\mathbf{s}^{ED}_{src,k} = \text{FFN}^{ED}(\mathbf{r}^{ED}_{src,k})\ \forall 1 \leq k \leq n$.

Next, the score vectors are sent to a Conditional Random Field (CRF) layer to learn the inter-dependencies between the tags and obtain conditional probability for possible tag sequences $P^{ED}(.|\mathbf{w} = x_{src})$. The negative log-likelihood of the golden tag sequence y_{src} is then used to train the model:

$$L^{ED} = -\sum_{(x_{src},y_{src})\in D_{src}} \log\big(P^{ED}(y_{src}|x_{src})\big) \tag{4.1}$$

Finally, Viterbi decoding is employed to perform prediction in inference time.

GATE (Ahmad et al., 2021): This is the current SOTA model for crosslingual RE and EAE on the ACE 2005 dataset. Given an input sentence $\mathbf{w}$ in x_{src}, this model uses the same encoding step with mBERT in BERTCRF to obtain the contextualized representation $\mathbf{z}_k$ for each $w_k \in \mathbf{w}$. Afterward, an overall word representation vector $\mathbf{v}_k$ for w_k is formed by the concatenation: $\mathbf{v}_k = [\mathbf{z}_k; \mathbf{z}^{pos}_k; \mathbf{z}^{dep}_k]$ where $\mathbf{z}^{pos}_k$ and $\mathbf{z}^{dep}_k$ are the embeddings of the universal part of speech and the dependency relation for w_k. Here, the dependency relation for a word is obtained by retrieving the dependency relation between the word and its governor in the dependency tree. For RE, given two entity mentions, the sequence of vectors $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n]$ is then passed to a Transformer layer (Vaswani et al., 2017) along with a syntax-based attention mask to compute a final representation vector $\mathbf{r}^{RE}_{src}$ for relation prediction over the input x_{src}. Afterward, a score vector for the possible relations is computed via a feed-forward network FFN^{RE}: $\mathbf{s}^{RE}_{src} = \text{FFN}^{RE}(\mathbf{r}^{RE}_{src})$.

The score vector $\mathbf{s}^{RE}_{src}$ is then sent to a softmax layer to obtain a distribution over possible relation types for x_{src}: $P^{RE}(.|x_{src})$. Finally, to train the model, we minimize the standard negative log-likelihood of the golden label y_{src}:

$$L^{RE} = -\sum_{(x_{src},y_{src})\in D_{src}} \log\big(P^{RE}(y_{src}|x_{src})\big) \tag{4.2}$$

For EAE, given an event trigger and an entity mention, we follow the same steps above for RE to compute the representation vector for role prediction $\mathbf{r}^{EAE}_{src}$, the score vector $\mathbf{s}^{EAE}_{src}$, and the negative log-likelihood for optimization L^{EAE}.

Finally, for convenience, let $\mathbf{r}^{ED}_{tgt,k}$, $\mathbf{r}^{RE}_{tgt}$, and $\mathbf{r}^{EAE}_{tgt}$ be the final representation vectors for x_{tgt} in the unlabeled data of target language. We also have $\mathbf{s}^{ED}_{tgt,k}$, $\mathbf{s}^{RE}_{tgt}$, and $\mathbf{s}^{EAE}_{tgt}$ for the likelihood score vectors for examples in the target language. These vectors are computed in the same way as their source language counterparts in this section.

4.1.3.2 Using Unlabeled Target Language Data. To avoid the monolingual bias in the cross-lingual methods for REE in Section 4.1.3.1, our work aims to exploit unlabeled data in the target language to improve the cross-lingual representations for REE. This section presents the typical approaches for leveraging unlabeled target language data for cross-lingual transfer learning in NLP, offering additional baselines for our proposed model later.

Language Adversarial Training (LADV): To leverage unlabeled data in the target language, this method introduces a language discriminator that receives representation vectors for input sentences and predicts the language identity (i.e., source or target) of the sentences (Cao, Liu, & Wan, 2020; X. Chen et al., 2019; Huang et al., 2019; Keung et al., 2019). As such, given an REE task

$t \in \{ED, RE, EAE\}$, the method seeks to jointly train a model for t (i.e., those described in Section 4.1.3.1) and the language discriminator so that the induced representation vectors for t can contain necessary information for the predictions in t and be language-agnostic to better transfer knowledge across languages at the same time.

To implement this method, we first obtain a representation vector for each input sentence in the source and target language data by feeding it into mBERT to obtain word representation vectors $[\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n]$ as in BERTCRF. Following (Keung et al., 2019), the average of such word vectors is used as the representation for the sentence in this baseline. For convenience, let $\mathbf{a}_{src}$ and $\mathbf{a}_{tgt}$ be the sentence representation vectors for the input sentences in x_{src} and x_{tgt} respectively. Also, let f^t_{lng} be the language discriminator for task t (implemented by a feed-forward network with a sigmoid activation in the end). In the next step, the representation vector $\mathbf{a}_*$ ($* \in \{src, tgt\}$) for each sentence is sent to f^t_{lng} to obtain a probability $p_* = f^t_{lng}(\mathbf{a}_*)$, indicating the likelihood that the input sentence belongs to the source language. Treating source and target language sentences as positive and negative examples, the loss for the discriminator L^{disc} is then computed via the negative log-likelihood: $L^{disc} = -\sum_{x_{src} \in D_{src}} \log(p_{x_{src}}) - \sum_{x_{tgt} \in D_{tgt}} \log\big(1 - p_{x_{tgt}}\big)$. The overall joint loss to train the model for t with LADV is thus: $L = L^{task} + L^{disc}$. Note that as LADV aims to prevent the language discriminator from recognizing the language identity from sentence representation vectors, we insert the Gradient Reversal Layer (GRL) (Ganin & Lempitsky, 2015) between $\mathbf{a}_*$ and f^{task}_{lng} to reverse the gradients during the backward pass from L^{disc}. Overall, fooling the language discriminator in LADV with GRL eliminates language-specific features to improve generalization across languages for t.

mBERT Finetuning (FMBERT): Recently, it has been shown that fine-tuning multilingual pre-trained language models on unlabeled data of the target language can improve the crosslingual performance for NLP tasks (Pfeiffer, Vulić, et al., 2020). Motivated by such prior work, this baseline exploits the unlabeled data in the target language for cross-lingual representation learning by fine-tuning mBERT on the data using mask language modeling (MLM) (Devlin et al., 2019a). Afterward, the fine-tuned mBERT model is utilized in the encoders for the baseline models for REE tasks in Section 4.1.3.1.

4.1.4 Proposed Method.

4.1.4.1 Class-based Alignment. An overview for the proposed model is shown in Figure 16. As described in the introduction, to avoid the potential cross-class alignment of representation vectors in the source and target language, this section presents a novel method for crosslingual representation alignment in REE where class information of tasks is explicitly employed to improve the alignment process. In particular, due to the language-universal nature of the semantics of the classes for an REE task, semantic representation vectors for a class should match each other no matter if they are computed with data from the source or target language. To this end, we seek to obtain two versions of representation vectors for each class in an REE task. One version is based on representations of examples for the source language while the other version employs representations from target language examples. The two representation versions will then be matched to achieve cross-lingual representation alignment for REE.

As such, let l be a class in an REE task t (e.g., l is a BIO tag for event types in ED). We compute the source-language representation $\mathbf{c}^t_{src,l}$ for l via the average of representation vectors for examples with label l in D_{src}. In particular, for $t = RE$ or EAE, we have:

$$\mathbf{c}^t_{src,l} = \frac{1}{N^l_{src}} \sum_{(x_{src},y_{src})} \nVdash[y_{src} = l]\mathbf{r}^t_{src} \tag{4.3}$$

Similarly, for $t = ED$:

$$\mathbf{c}^{ED}_{src,l} = \frac{1}{N^l_{src}} \sum_{(x_{src},y_{src})} \sum_{k=1}^{|x_{src}|} \nVdash[y_{src,k} = l]\mathbf{r}^{ED}_{src,k} \tag{4.4}$$

Here, $\nVdash$ is the indicator function, and N^l_{src} is the number of examples (for RE and EAE) or words (for ED) in D_{src} that are annotated with label l.

In the target language, as the golden labels y_{tgt} for the examples x_{tgt} are not provided, we propose to obtain a target-language representation $\mathbf{c}^t_{tgt,l}$ by aggregating representation vectors for all examples $x_{tgt} \in D_{tgt}$. Probability estimations for examples or words to belong to class l are used as the weights for the aggregation. In particular, we obtain the probability estimations by sending the score vectors $\mathbf{s}^{ED}_{tgt,k}$, $\mathbf{s}^{RE}_{tgt}$, and $\mathbf{s}^{EAE}_{tgt}$ to a softmax layer: $\hat{\mathbf{y}}^{ED}_{tgt,k} = \text{softmax}(\mathbf{s}^{ED}_{tgt,k})$, and $\hat{\mathbf{y}}^t_{tgt} = \text{softmax}(\mathbf{s}^t_{tgt})$ (for $t = RE$ or EAE). As such, we obtain the target-language representation for l via the weighted sum of $\mathbf{r}^t_{tgt}$ (for RE and EAE):

$$\mathbf{c}^t_{tgt,l} = \frac{\sum_{x_{tgt} \in D_{tgt}} \hat{\mathbf{y}}^t_{tgt,l}\mathbf{r}^t_{tgt}}{\sum_{x_{tgt} \in D_{tgt}} \hat{\mathbf{y}}^t_{tgt,l}} \tag{4.5}$$

Similarly, for ED:

$$\mathbf{c}^{ED}_{tgt,l} = \frac{\sum_{x_{tgt} \in D_{tgt}} \sum_{k=1}^{|x_{tgt}|} \hat{\mathbf{y}}^{ED}_{tgt,k,l}\mathbf{r}^{ED}_{tgt,k}}{\sum_{x_{tgt} \in D_{tgt}} \sum_{k=1}^{|x_{tgt}|} \hat{\mathbf{y}}^{ED}_{tgt,k,l}} \tag{4.6}$$

where $\hat{\mathbf{y}}^t_{tgt,l}$ and $\hat{\mathbf{y}}^{ED}_{tgt,k,l}$ represent the likelihood score for class l in vectors $\hat{\mathbf{y}}^t_{tgt}$ and $\hat{\mathbf{y}}^{ED}_{tgt,k}$ respectively. The alignment for the representations of class l is then achieved by minimizing the negative cosine similarity of the source- and target-language vectors (i.e., for task t):

$$L^t_{cls} = -\sum_l \text{cosine}(\mathbf{c}^t_{src,l}, \mathbf{c}^t_{tgt,l}) \tag{4.7}$$

Adaptive Coefficient: In our implementation, we compute the source-language representations $\mathbf{c}^t_{src,l}$ for l after each training epoch while the target-language representations $\mathbf{c}_{tgt,l}^t$ are obtained for in each training minibatch. The current parameters of the models are utilized to perform such calculation. As such, the quality of the representation vectors for classes might vary along the training process of the models. In particular, later epochs might correspond to better model parameters, thus leading to more reliable class representations. To this end, we propose to apply an adaptive coefficient λ_{cls} for the class alignment loss L_{cls}^t so its impact is gradually increasing along the training: $\lambda_{cls} = \frac{2}{1+\exp(-e/E)} - 1$ where E and e are the total and current numbers of training epochs, respectively. Note that λ_{cls} is small in the early training stages and gradually increase in the process.

4.1.4.2 Word Category-based Alignment. We further exploit universal parts of speech (UPOS) and dependency relations as the language-agnostic knowledge to align crosslingual representations for REE. To achieve a fair comparison with prior work (Ahmad et al., 2021; Subburathinam et al., 2019), we employ the UDPipe toolkit (Straka & Straková, 2017) to obtain parts of speech and dependency relations for the sentences. Due to their similarity, we will only describe the UPOS-based alignment process and the dependency-based alignment can be done in the same way.

As such, we utilize an embedding table U (initialized randomly) to capture representation vectors for the possible UPOS, serving as an anchor knowledge across languages. Next, to facilitate the UPOS-based representation alignment, we compute additional representation vectors for UPOS based on representation vectors of examples in both source and target languages. In particular, for each word w_k in an input sentence $\mathbf{w}$ (from x_{src} or x_{tgt}), we send its contextualized representation $\mathbf{z}_k$ from mBERT into a feed-forward network FFN^{UPOS} to produce a representation vector $\mathbf{q}_k$ for the UPOS w_k^{pos} of $w_k \in \mathbf{w}$: $\mathbf{q}_k = \text{FFN}^{UPOS}(\mathbf{z}_k)$.

Afterward, to leverage the language-universal of U, we propose to match $\mathbf{q}_k$ to the embedding vector of w_k^{pos} in U for $\mathbf{q}_k$ in both source and target language data. In other words, induced representation vectors in the source and target languages are both matched to the anchor knowledge U, providing a mechanism to align source and target representations.

To match $\mathbf{q}_k$ and U, we seek to maximize the similarity between $\mathbf{q}_k$ and the embedding of w_k^{pos} in U while minimizing $\mathbf{q}_k$'s similarities with embeddings of other UPOS at the same time. To implement this idea, we utilize the following function for minimization:

$$L_{pos}^{align} = \sum_{\mathbf{w} \in D, w_k \in \mathbf{w}} \log \left(\sum_{u \in O} e^{\mathbf{q}_k U[u] - \mathbf{q}_k U[w_k^{pos}]} \right) \tag{4.8}$$

where $D = D_{src} \cup D_{tgt}$, O is the set of possible UPOS, and $U[u]$ is the embedding of u in U.

Context Information Filtering: Note that L_{pos}^{align} is also the negative log-likelihood for a feed-forward classifier that uses U as the weight matrix and $\mathbf{q}_k$ as the input vector to predict the UPOS w_k^{pos} for w_k. As such, minimizing L_{pos}^{align} also serves to retain relevant information for UPOS prediction in the representation vector $\mathbf{q}_k$. However, due to the direct computation of $\mathbf{q}_k$ from the contextualized representation $\mathbf{z}_k$, it is possible that $\mathbf{q}_k$ still preserves context information from the input sentence $\mathbf{w}$. This might introduce noise into $\mathbf{q}_k$ as ideally, we expect $\mathbf{q}_k$ to focus only on information about UPOS. As such, to improve the quality of $\mathbf{q}_k$ for representation alignment, we propose to explicitly filter context information from vectors $\mathbf{q}_k$. Our main idea is to ensure that $\mathbf{q}_k$ cannot be used to recover the context words in $\mathbf{w}$. To achieve this goal, we first obtain an aggregated vector for the UPOS representation vectors in the input sentence $\mathbf{w}$: $\overline{\mathbf{q}} = \frac{1}{n} \sum_{k=1}^{n} \mathbf{q}_k$. The resulting vector is then fed into a Gradient Reversal Layer (GRL) (Ganin &

Lempitsky, 2015), followed by a word classifier (i.e., a feed-forward network FFN^{ctx} with a softmax layer in the end) to compute a probability distribution over the words in our vocabulary: $\hat{\mathbf{y}}^{ctx} = \text{softmax}(\text{FFN}^{ctx}(\text{GRL}(\overline{\mathbf{q}})))$. Finally, to filter the context information from $\mathbf{q}_k$, we minimize the negative log-likelihood of the context words w_k in the input sentence $\mathbf{w}$:

$$L_{pos}^{ctx} = - \sum_{\mathbf{w} \in D_{src} \cup D_{tgt}} \sum_{w_k \in \mathbf{w}} \log\big(\hat{\mathbf{y}}^{ctx}[w_k]\big) \tag{4.9}$$

where $\hat{\mathbf{y}}^{ctx}[w_k]$ is the probability for word w_i in the distribution $\hat{\mathbf{y}}^{ctx}$. Note that while the minimization of the negative log-likelihood generally encourages input representations to reveal information about the prediction outputs (i.e., context words in our case), the introduction of GRL in L_{pos}^{ctx} reverses this process to discourage the context information in $\overline{\mathbf{q}}$, thus purifying $\mathbf{q}_k$ to focus on UPOS knowledge and facilitating the representation alignment.

In the next steps for universal dependency relations, we follow the same procedure for L_{pos}^{align} and L_{pos}^{ctx} to obtain the losses L_{dep}^{align} and L_{dep}^{ctx} respectively for minimization. For convenience, let $L_{pos} = L_{pos}^{align} + L_{pos}^{ctx}$ and $L_{dep} = L_{dep}^{align} + L_{dep}^{ctx}$. In summary, the overall loss function to train our models for a task $t \in \{ED, RE, EAE\}$ with both class and word category alignment is thus: $L^{main} = L^t + \lambda_{cls} L_{cls}^t + \lambda_{pos} L_{pos} + \lambda_{dep} L_{dep}$ where λ_{cls} is the adaptive coefficient, and λ_{pos} and λ_{dep} are trade-off parameters.

4.1.5 Experiments.

Datasets and Hyper-parameters: Following previous work (Ahmad et al., 2021; M'hamdi et al., 2019; Subburathinam et al., 2019), we use the multilingual dataset ACE 2005 (Walker et al., 2006) to evaluate REE models in this work. ACE 2005 annotate documents for entity mentions, event triggers, relations, and arguments in English (EN), Chinese (ZH) and Arabic (AR). We apply the same data split and preprocessing for ACE 2005 as prior work

Language	Data	RE (**#rels**)	ED (**#trgs**)	EAE (**#args**)
English	Train	4,974	4,420	7,018
	Dev	626	505	877
	Test	620	424	878
Chinese	Train	4,767	2,213	5,931
	Dev	572	111	741
	Test	605	197	742
Arabic	Train	2,918	1,986	3,959
	Dev	357	112	495
	Test	378	169	495

Table 21. Statistics of the multilingual datasets for ED, RE, and EAE in ACE 2005. **#rels**, **#trgs** and **#args** represent the numbers of relations, event triggers, and event arguments respectively.

Model	Even Argument Extraction						Relation Extraction					
	EN ZH	EN AR	ZH EN	ZH AR	AR EN	AR ZH	EN ZH	EN AR	ZH EN	ZH AR	AR EN	AR ZH
GATE	63.2	68.5	59.3	69.2	53.9	57.8	55.1	66.8	71.5	61.2	69.0	54.3
GATE+LADV	63.9	67.7	60.3	68.6	55.8	57.8	56.8	64.2	70.2	61.6	68.9	54.8
GATE+FMBERT	63.7	68.7	59.3	**69.3**	54.6	58.1	55.8	66.9	71.8	61.7	69.2	54.9
GATE+CCCAR	**65.5**	**69.4**	**62.0**	**69.3**	**57.5**	**59.1**	**58.1**	**67.9**	**72.0**	**63.5**	**70.5**	**57.7**

Table 22. Performance (F1 scores) of models on test data for EAE and RE in six crosslingual settings. Each column corresponds to one setting where source languages are written above target languages. Underlined numbers designate settings where the proposed model is significantly better than other models with $p < 0.01$.

(Ahmad et al., 2021; M'hamdi et al., 2019) for a fair comparison. Overall, there are 18 relation types, 33 event types, and 35 argument roles in this dataset. For each of the language (i.e., English, Chinese and Arabic) and task (i.e., ED, RE, and EAE), the data split provides training, development, and test data. In our cross-lingual transfer learning experiments, the models will be trained on the training data of one language (the source) and evaluated on the test data of another language (the target). The unlabeled data for the target language is obtained by removing the labels from its training data. The statistics of the ACE 2005 dataset for the three tasks are shown in Table 21.

We use the same hyper-parameters for BERTCRF and GATE as provided by previous work (Ahmad et al., 2021; M'hamdi et al., 2019). Specific hyper-

Model	Event Detection					
	EN ZH	EN AR	ZH EN	ZH AR	AR EN	AR ZH
BERTCRF	68.5	30.9	-	-	-	-
BERTCRF+LADV	70.0	33.5	41.2	20.3	37.2	55.6
BERTCRF+FMBERT	69.4	33.4	42.9	20.0	36.5	56.3
BERTCRF+CCCAR	**72.1**	**42.7**	**45.8**	**20.7**	**40.7**	**59.8**

Table 23. Performance (F1 scores) on test data for ED in six crosslingual settings. Each column corresponds to one setting where source languages are written above target languages. "-" indicates results that are not reported in the original work. Underlined numbers designate settings where the proposed model is significantly better than other models with $p < 0.01$.

parameters for our model are tuned on the development data. In particular, we use two layers for the feed forward networks with 50 hidden units for the layers, 50 dimensions for the UPOS and dependency embeddings, and 0.1 for the parameters λ_{pos} and λ_{dep}. For the baseline FMBERT, we utilize the *huggingface* library to finetune mBERT on unlabeled target data with MLM for $100,000$ steps (i.e., batch size of 64 and learning rate of $5e$-5).

Performance Comparison: We compare the proposed crosslingual method for REE on two groups of baselines. The first group involve models that only use source language data for training, i.e., BERTCRF and GATE. These are current SOTA methods for crosslingual ED, RE, and EAE. The second baseline groups additionally employ unlabeled data in the target language to support crosslingual representation learning in REE, i.e., LADV and FMBERT. Our proposed method also leverages unlabeled data in the target language, called CCCAR for class- and word category-based crosslingual alignment of representations. Note that LADV, FMBERT, and CCCAR should be applied on top of a source-only method (i.e., BERTCRF and GATE) to form a complete model.

Tables 23 and 22 show the test data performance of the models for the three REE tasks in six crosslingual settings (i.e., with different pairs of languages for the source and target). It is clear from the tables that the proposed method CCCAR consistently outperforms other methods in all crosslingual settings for the three REE tasks. In particular, for EAE, CCCAR substantially improves the baseline model GATE (i.e., the current SOTA) by 1.9% on average while those improvement for LADV and FMBERT are only 0.45% and 0.38%. The same trend can be seen for RE and ED where CCCAR on average improves the baselines by 1.97% for the former and 7.7% for the latter. These results clearly demonstrate the effectiveness of the proposed method, highlighting the benefits of the class- and word category-based alignment for crosslingual REE.

Model	English → Chinese			English → Arabic		
	RE	ED	EAE	RE	ED	EAE
CCCAR	**58.1**	**72.1**	**65.5**	**67.9**	**42.7**	**69.4**
- Class Align.	56.6	69.9	63.6	66.9	38.8	68.9
- Adaptive Coeff.	57.4	71.5	64.7	67.3	41.3	69.2
- UPOS Align.	57.9	71.4	65.1	66.9	40.4	69.3
- Dep Align.	57.8	71.7	64.7	67.1	41.5	68.9
- Word Cat Align.	57.0	70.9	64.4	67.0	40.0	68.7
- Context Filtering	57.6	71.2	64.9	67.4	41.6	69.0

Table 24. Performance (F1 scores) of models. In the row for the proposed model CCCAR, we use BERTCRF as the base model for ED, and GATE as the base model for RE and EAE.

Ablation Study: This section conducts an ablation study to understand the contribution of each designed component in the proposed crosslingual alignment method CCCAR. In particular, we examine the performance of the following ablated models: (i) **- Class Align.**: this model excludes the class-based alignment component (i.e., the loss L^t_{cls}) from CCCAR; (ii) **- Adaptive Coeff.**: instead of using the adaptive coefficient λ_{cls} for the class-based alignment loss L^t_{cls}, this model utilizes a fixed value (i.e., 0.2 as tuned on development data) for λ_{cls}; (iii) **- UPOS Align.**: this model eliminates the UPOS-based alignment component (i.e., the losses L^{align}_{pos} and L^{ctx}_{pos}) from CCCAR; (iv) **- Dep Align.**: the alignment component

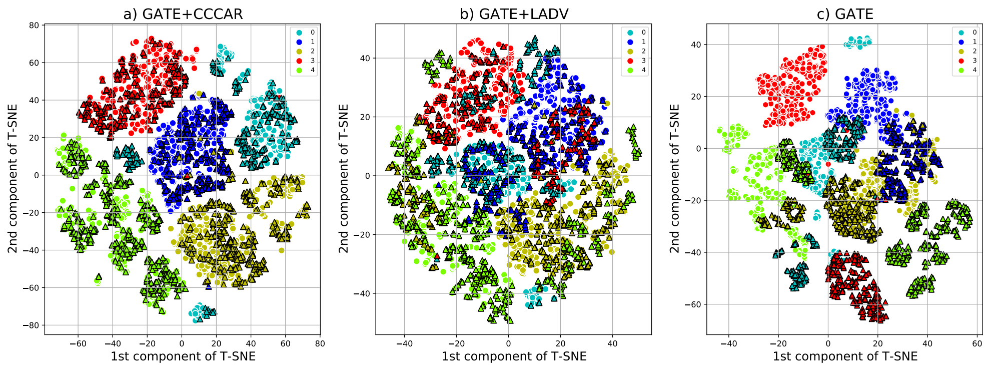


Figure 17. T-SNE visualizations for the representations of 4,000 randomly selected examples from English (i.e., source language) and Chinese (i.e., target language) data. Circles and triangles represent English and Chinese examples respectively. Colors represent different classes in EAE. GATE+CCCAR shows induced representation vectors from our proposed model.

based on dependency relations (i.e., the losses L_{dep}^{align} and L_{dep}^{ctx}) is not utilized in this model; (v) **- Word Cat Align.**: this model removes both UPOS-based and dependency-based alignment from CCCAR (i.e., excluding L_{pos} and L_{dep}); and (vi) **- Context Filtering**: the word context filtering for the representation vectors of UPOS and dependency relations (with GRL) is not employed in this model (i.e., eliminating the losses L_{pos}^{ctx} and L_{dep}^{ctx}).

Table 24 presents the test data performance of the models in the English-to-Chinese and English-to-Arabic settings for the three REE tasks. As can be seen, removing any component of the proposed model would hurt the performance significantly across different settings and tasks, thus clearly illustrating the benefits of the designed components for CCCAR. The performance of the models drops the most when the class-based alignment is excluded, further demonstrating the importance of class-aware alignment for crosslingual REE.

Source-language Data Usage: Previous experiments show that using unlabeled data in the target language to align representation vectors in CCCAR can improve

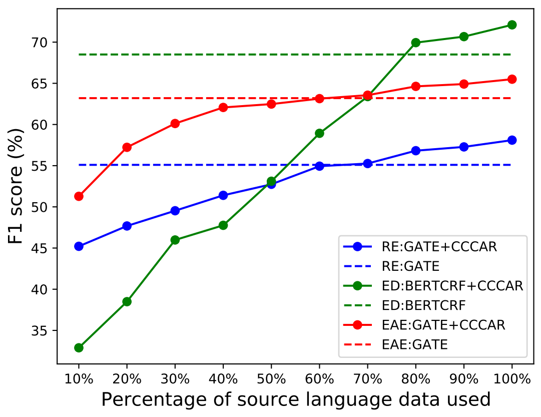


Figure 18. Performance on test data of the models in the English-to-Chinese setting. Dash lines represent the performance of the source-only baselines using 100% of the source-language training data.

the performance for the source-only baselines for REE. In this section, we seek to understand how much labeled data in the source language can be saved if unlabeled data in the target language is employed with CCCAR for an REE task. In particular, we are interested in the portion of source language data that, once combined with unlabeled target language data via CCCAR, can produce similar performance as the source-only baseline trained on full source language data. To this end, we show the learning curves of the source-only and CCCAR-augmented models for REE tasks when the size of the source-language training data varies. Figure 18 show the curves for the English-to-Chinese setting. As can be seen, the proposed CCCAR method with unlabeled target data only needs to use approximately 60% of the source-language training data for RE and EAE to achieve comparable performance with the source-only baselines on full source language data. This portion for ED is less than 80%. These results thus suggests an additional benefit of CCCAR to significantly reduce necessary data annotation for the source language based on unlabeled target language data in crosslingual learning for REE.

Alignment Effect of the Proposed Method: As discussed earlier, a major issue for LADV is that it might align representations of examples with different classes in the crosslingual setting. CCCAR can address this issue as it explicitly relies on class information for representation alignment. To demonstrate these arguments, Figure 17 uses the t-Distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten & Hinton, 2008) to visualize the example representations induced by GATE, the LADV baseline GATE+LADV, and the proposed GATE+CCCAR. This visualization is done over 4,000 randomly selected examples for the top 5 frequent classes in EAE. Here, examples are sampled from training data for both source and target languages in the English-to-Chinese setting. As can be seen, in the source-only model GATE, representations for examples from the source language are quite separate from those in the target language. The representation alignment in GATE+LADV can address this issue by pushing representations from both languages closer. However, representations for examples with different classes are unexpectedly aligned in GATE+LADV, causing suboptimal representations for crosslingual settings. Finally, due to the explicit condition on class information for alignment, GATE+CCCAR can match representations for both languages while avoiding the cross-class alignment to improve crosslingual performance for REE.

4.1.6 Related Work.

REE has been extensively studied for English, featuring traditional machine learning methods (Q. Li et al., 2013a; Liao & Grishman, 2011; Patwardhan & Riloff, 2009; B. Yang & Mitchell, 2016a) and advanced deep learning models (Y. Chen, Xu, Liu, Zeng, & Zhao, 2015b; Y. Lin et al., 2020a; M. V. Nguyen, Lai, & Nguyen, 2021; T. H. Nguyen, Cho, & Grishman, 2016a; T. H. Nguyen & Grishman, 2015a, 2018b; Sahu, Christopoulou, Miwa, & Ananiadou, 2019; Veyseh, Dernoncourt, Dou, & Nguyen, 2020b; Veyseh, Dernoncourt, Thai, Dou, & Nguyen, 2020b; Veyseh, Nguyen, & Nguyen, 2020b; X. Wang et al., 2019; Zhang et al., 2019). Recently, several works have considered cross-lingual transfer learning for three REE tasks (J. Liu et al., 2019a; Ni & Florian, 2019; Subburathinam et al., 2019) where multilingual pre-trained language models (e.g., mBERT) have been proved as an important encoding component (Ahmad et al., 2021; M. V. Nguyen & Nguyen, 2021b).

However, a fundamental limitation of existing crosslingual models for REE is the monolingual bias due to the sole reliance on source language data for training. In other NLP tasks, LADV has been explored to address this issue by leveraging unlabeled data in the target language to perform crosslingual representation alignment (Cao et al., 2020; X. Chen et al., 2019; He et al., 2020; Huang et al., 2019; Lange et al., 2020a). Unfortunately, LADV suffers from the cross-class alignment issue, making it less optimal for crosslingual REE. Finally, we note that language-universal representation learning is related to domain adaption research where models seek to learn domain-invariant representations (Adel, Zhao, & Wong, 2017; Cicek & Soatto, 2019; L. Fu, Nguyen, Min, & Grishman, 2017; Ganin & Lempitsky, 2015; Ngo Trung, Phung, & Nguyen, 2021; Tang, Chen, & Jia, 2020; Xie, Zheng, Chen, & Chen, 2018).

4.1.7 Summary. We present a novel method for crosslingual transfer learning for REE that leverages unlabeled data in the target language to support language-universal representation learning. Our method exploits class semantics in REE tasks and universal word categories (i.e., UPOS and dependency relations) as bridges to align representation vectors across languages. In our method, representation vectors for classes and word categories are computed via contextualized representations of examples to implement representation matching for crosslingual alignment. Extensive experiments show that the proposed method achieves SOTA performance for three REE tasks in different crosslingual settings.

4.2 FAMIE

4.2.1 Introduction.

Information Extraction (IE) systems provide important tools to extract structured information from text (V. D. Lai, Nguyen, Nguyen, & Dernoncourt, 2021; Q. Li et al., 2014; M. V. Nguyen, Lai, & Nguyen, 2021; T. M. Nguyen & Nguyen, 2019b; Veyseh, Nguyen, Min, & Nguyen, 2021). At the core of IE involves sequence labeling tasks that aim to recognize word spans and semantic types for some objects of interest (e.g., entities and events) in text. For example, two typical sequence labeling tasks in IE feature Named Entity Recognition (NER) to find names of entities of interest, and Event Detection (ED) to identify triggers of specified event types (Walker et al., 2006). Despite extensive research effort for sequence labeling (Lafferty et al., 2001; Ma & Hovy, 2016; Pouran Ben Veyseh, Nguyen, Ngo Trung, Min, & Nguyen, 2021), a major bottleneck of existing IE methods involves the requirement for large-scale human-annotated data to build high-quality models. As annotating data is often expensive and time-consuming, large-scale labeled data is not practical for various domains and languages.

To address the annotation cost for IE, previous work has resorted to active learning (AL) approaches (Settles, 2009; Settles & Craven, 2008) where only a selective set of examples are annotated to minimize the annotation effort while maximizing the performance. Starting with a set of unlabeled data, AL methods train and improve a sequence labeling model via multiple human-model collaboration iterations. At each iteration, three major steps are performed in order: (i) training the model on the current labeled data, (ii) using the trained model to select the most informative examples in the current unlabeled set for annotation, and (iii) presenting the selected examples to human annotators to obtain labels. In AL, the number of annotated samples or annotation time might be limited by a budget to make it realistic.

Unfortunately, despite much potentials, existing AL methods and frameworks are still not applied widely in practice due to their main focus on devising the most effective example selection algorithm for human annotation, e.g., based on the diversity of the examples (Shen, Yun, Lipton, Kronrod, & Anandkumar, 2017a; M. Yuan, Lin, & Boyd-Graber, 2020) and/or the uncertainty of the models (Roth & Small, 2006; Shelmanov et al., 2021; D. Wang & Shang, 2014). Training and selection time in the first and second steps of each AL interaction is thus not considered in prior work for sequence labeling. This is a critical issue that limits the application of AL: annotators might need to wait for a long period between annotation batches due to the long training and selection time of the models at each AL iteration. Given the widespread trend of using large-scale pre-trained language models (e.g., BERT), this problem of long waiting or training/selection time in AL can only become worse. On the one hand, the long idle time of annotators reduces the number of annotated examples given an annotation budget. Further, the engagement of annotators in the annotation process can drop significantly due to the long interruptions between annotation rounds, potentially affecting the quality of their produced annotation. In all, current AL frameworks are unable to optimize the available time of annotators to maximize the annotation quantity and quality for satisfactory performance.

To this end, we demonstrate a novel AL framework (called FAMIE) that leverages large-scale pre-trained language models for sequence labeling to achieve optimal modeling capacity while significantly reducing the waiting time between annotation rounds to optimize annotator time. Instead of training the full/main large-scale model for data selection at each AL iteration, our key idea is to train only a small proxy model on the current labeled data to recommend new examples for annotation in the next round. In this way, the training and data selection time can be reduced significantly to enhance annotation engagement and quality. An important issue in this idea is to ensure that the examples selected by the proxy model are also optimal for the main large model. To this end, we introduce a novel knowledge distillation mechanism for AL that encourages the synchronization between the proxy and main models, and promotes the fitness of selected examples for the main model. To update the main model with new annotated data for effective distillation, we propose to train the main large model on current labeled data during the annotation time, thus not adding to the waiting time of annotators between annotation rounds. This is in contrast to previous AL frameworks that leave the computing resources unused during annotation time. Our approach can thus efficiently exploit both human and computer time for AL.

To evaluate the proposed AL framework FAMIE, we conduct experiments for multilingual sequence labeling problems, covering two important IE tasks (i.e., NER and ED) in three languages (i.e., English, Spanish, and Chinese). The experiments demonstrate the efficiency and effectiveness of FAMIE that can achieve strong performance with significantly less human-computer collaboration time. Compared to existing AL systems such as ActiveAnno (Wiechmann, Yimam, & Biemann, 2021) and Paladin (Nghiem, Baylis, & Ananiadou, 2021), our system

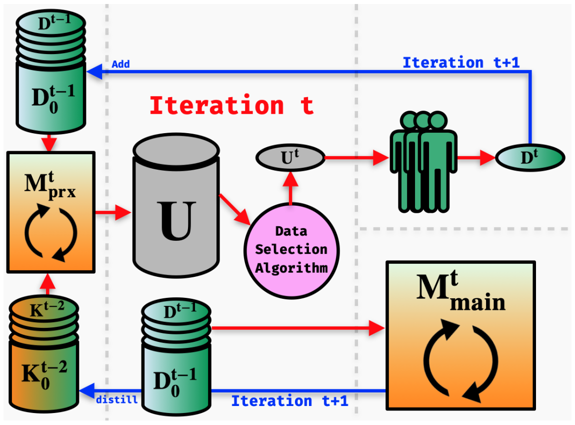


Figure 19. The overall Proxy Active Learning process.

FAMIE features important advantages. First, FAMIE introduces a novel approach to reduce model training and data selection time for AL via a small proxy model and knowledge distillation while still benefiting from the advances in large-scale language models. Second, while previous AL systems only focus on some specific task in English, FAMIE can support different sequence labeling tasks in multiple languages due to the integration of our prior multilingual toolkit Trankit (M. V. Nguyen, Lai, Veyseh, & Nguyen, 2021) to perform fundamental NLP tasks in 56 languages. Third, in contrast to previous AL systems that only implement one data selection algorithm, FAMIE covers a diverse set of AL algorithms. Finally, FAMIE is the first complete AL system that allows users to define their sequence labeling problems, work with the models to annotate data, and eventually obtain a ready-to-use model for deployment.

4.2.2 System Description. In AL, we are given two initial datasets, a small seed set of labeled examples $D_0 = \{(\mathbf{w}, \mathbf{y})\}$ and an unlabeled example set $U_0 = \{\mathbf{w}\}$ (the seed set D_0 is optional and our system can work directly with only U_0). For sequence labeling, models consume a sequence of K words $\mathbf{w} = [w_1, w_2, \ldots, w_K]$ (i.e., a sentence/example) to output a tag sequence

$\mathbf{y} = [y_1, y_2, \ldots, y_K]$ (y_i is the label tag for w_i). The tag sequence is represented in the BIO scheme to capture spans and types of objects of interest.

A typical AL process contains multiple rounds/iterations of model training, data selection, and human annotation in a sequential manner. Let D and U be the overall labeled and unlabeled set of examples at the beginning of the current t-th iteration (initialized with D_0 and U_0). At the current iteration, a sequence labeling model is first trained on the current labeled set D. A sample selection algorithm then employs the trained model to suggest the most informative subset of examples U^t in U (i.e., $U^t \subset U$) for annotation. Afterwards, a human annotator will provide labels for the sentences in the selected set U^t, leading to the labeled examples D^t for U^t. The labeled and unlabeled sets can then be updated via: $D \leftarrow D \cup D^t$ and $U \leftarrow U \setminus U^t$.

4.2.2.1 Model. We employ the typical Transformer-CRF architecture for sequence labeling (M. V. Nguyen, Lai, Veyseh, & Nguyen, 2021). In particular, given the input sentence $\mathbf{w} = [w_1, w_2, \ldots, w_K]$, the state-of-the-art multilingual language model XLM-Roberta (Conneau et al., 2020) is used to obtain contextualized embeddings for the words: $\mathbf{X} = \mathbf{x}_1, \ldots, \mathbf{x}_K = \text{XLMR}(w_1, \ldots, w_K)$ (i.e., to support multiple languages). Afterwards, the word embeddings are sent to a feed-forward network with softmax in the end to obtain the score vectors: $\mathbf{z}_i = \text{softmax}(\mathbf{h}_i)$ where $\mathbf{h}_i = \text{FFN}(\mathbf{x}_i)$. Here, each value in $\mathbf{z}_i$ represents a score for a tag in the tag set V. The score vectors are then fed into a Conditional Random Field (CRF) layer to compute a distribution for possible tag sequences for $\mathbf{w}$: $P(\hat{\mathbf{y}}|\mathbf{w}) = \frac{\exp(s(\hat{\mathbf{y}},\mathbf{w}))}{\sum_{\hat{\mathbf{y}}' \in Y(\mathbf{w})} \exp(s(\hat{\mathbf{y}}',\mathbf{w}))}$ where $Y(\mathbf{w})$ is the set of all possible tag sequences for $\mathbf{w}$. Also, $s(\hat{\mathbf{y}}, \mathbf{w})$ is the score for a tag sequence $\hat{\mathbf{y}} = [\hat{y}_1, \ldots, \hat{y}_K]$: $s(\hat{\mathbf{y}}, \mathbf{w}) = \sum_i \mathbf{z}_i[\hat{y}_i] + \sum_i \pi_{\hat{y}_i \to \hat{y}_{i+1}}$. Here, $\pi_{\hat{y}_i \to \hat{y}_{i+1}}$ is the transition score from the tag

$\hat{y}_i$ to the tag $\hat{y}_{i+1}$. The model is trained by minimizing the negative log likelihood: $L_{task} = -\log P(\mathbf{y}|\mathbf{w})$. For inference, the Viterbi algorithm is used for decoding: $\hat{\mathbf{y}^*} = \max_{\hat{\mathbf{y}}'} P(\hat{\mathbf{y}}'|\mathbf{w})$.

Adapter-based Finetuning To further improve the memory and time efficiency, we incorporate light-weight adapter networks (Houlsby et al., 2019; Peters, Ruder, & Smith, 2019a) into our model. In form of small feed-forward networks, adapters are injected in between the transformer layers of XLM-Roberta. For training, we only update the adapters while the parameters of XLM-Roberta are fixed. This significantly reduces the amount of learning parameters while sacrificing minimal extraction loss, or in case of low-resource learning even surpassing performance of fully fine-tuned models.

4.2.2.2 Data Selection Strategies. To improve the flexibility to accommodate different problems, our AL framework supports a wide range of data selection strategies for choosing the best batch of examples to label at each iteration for sequence labeling. These algorithms are categorized into three groups, i.e., uncertainty-based, diversity-based, and hybrid. For each group, we explore its most popular methods as follows.

Uncertainty-based. These methods select examples for annotation according to the main model's confidence over the predicted tag sequences for unlabeled examples. Early methods sort the unlabeled examples by the uncertainty of the main model. To avoid the preference over longer examples, the method Maximum Normalized Log-Probability (**MNLP**) (Shen et al., 2017a) proposes to normalize the likelihood over example lengths. In particular, MNLP selects examples with the highest MNLP scores: $MNLP(\mathbf{w}) = -\max_{\hat{\mathbf{y}}'} \frac{1}{K} \log P(\hat{\mathbf{y}}'|\mathbf{w})$.

Diversity-based. Algorithms in this category assume that a representative set of examples can act as a good surrogate for the whole dataset. **BERT-KM** (M. Yuan et al., 2020) uses K-Means to cluster the examples in unlabeled data based on the contextualized embeddings of the sentences (i.e., the representations for the [CLS] tokens in the trained BERT-based models). The nearest neighbors to the K cluster centers are then chosen for labeling.

Hybrid. Recently, several works have proposed data selection strategies for BERT-based AL to balance between uncertainty and diversity. The **BADGE** method (Ash, Zhang, Krishnamurthy, Langford, & Agarwal, 2019; Kim, 2020) chooses examples from clusters of gradient embeddings, which are formed with the token representations $\mathbf{h}_i$ from the penultimate layer of the main model and the gradients of the cross-entropy loss with respect to such token representations. The gradient embeddings are then sent to the K-Means++ to find the initial K cluster centers that are distant from each other, serving as the selected examples (Kim, 2020).

In addition, we implement the AL framework **ALPS** (M. Yuan et al., 2020) that does not require training the main model for data section. ALPS employs the surprisal embedding of $\mathbf{w}$, which is obtained from the likelihoods of masked tokens from pre-trained language models (i.e., XLM-Roberta). The surprisal embeddings are also clustered to select annotation examples as in BERT-KM.

4.2.2.3 Proxy Active Learning. As discussed in the introduction, model training and data selection at each iteration of traditional AL methods might consume significant time (especially with the current trend of large-scale language models), thus introducing a long idle time for annotators that might reduce annotation quality and quantity. To this end, (Shelmanov et al., 2021) have explored approaches to accelerate training and data selection steps for AL by leveraging smaller and approximate models during the AL iterations. To make it more efficient, the main large model is only trained once in the end over all the annotated examples in AL. Unfortunately, this approach suffers from the mismatch between the approximate and main models as they are separately trained in AL, thus limiting the effectiveness of the selected examples for the main model (Lowell, Lipton, & Wallace, 2019).

To overcome these issues, our AL framework FAMIE trains a small proxy network at each iteration to suggest new unlabeled samples. Dealing with the mismatch between the proxy-selected examples and the main model, FAMIE proposes to involve the main model in the training and data selection for the proxy model. In particular, at each AL iteration, the main model will still be trained over the latest labeled data. However, to avoid the interference of the main large model with the waiting time of annotators, we propose to train the main model during the annotation time of the annotators (i.e., main model training and data annotation are done in parallel). Given the main model trained at previous iteration, knowledge distillation will be employed to synchronize the knowledge between the main and proxy models at the current iteration.

The complete framework for FAMIE is presented in Figure 19. At iteration t, a proxy acquisition model is trained on the current labeled data set $D_0^{t-1} = D^0 \cup D^1 \ldots \cup D^{t-1}$. The trained proxy model at the current step is called M_{prx}^t. Also, we use knowledge distillation signals K_0^{t-2} that is computed from the main model M_{main}^{t-1} trained at the previous iteration $t-1$ to synchronize the proxy model M_{prx}^t and the main model M_{main}^{t-1} (M_{prx}^1 is trained only on D^0). Afterwards, a data selection algorithm is used to select a batch of examples U^t from the current unlabeled set U for annotation, leveraging the feedback from M^t_{prx}. Next, a human annotator will label U^t to produce the labeled data batch D^t for the next iteration $t+1$. During this annotation time, the main model will also be trained again over the current labeled data D^{t-1}_0 to produce the current version M^t_{main} of the model. The distillation signal K^{t-1}_0 for the next step will also be computed after the training of M^t_{main}. This process is repeated over multiple iterations and the last version of M_{main} will be returned for users.

To improve the fitness of the proxy-based selected examples for M_{main}, we leverage the distilled version miniLM of XLM-Roberta (W. Wang, Bao, Huang, Dong, & Wei, 2021) that employs similar stacks of transformer layers for the proxy model M_{prx}. Note that M_{prx} also includes a CRF layer on top of miniLM.

4.2.2.4 Uncertainty Distillation. Although the proxy and main model M_{prx} and M_{main} are trained on similar data, they might still exhibit large mismatch, e.g., regarding decision boundaries. This prompts a demand for regularizing the proxy model's predictions to be consistent with those of a trained main model to improve the fitness of the selected examples for M_{main}. Ideally, we expect the tag sequence distribution $P_{prx}(\mathbf{y}|\mathbf{w})$ learned by the proxy model to mimic the tag sequence distribution $P_{main}(\mathbf{y}|\mathbf{w})$ learned by the main model. To this end, we propose to minimize the difference between the intermediate outcomes (i.e., the unary and transition scores) of the two distributions. In particular, we introduce the following distillation objective for each sentence $\mathbf{w}$ at one AL iteration: $L_{dist} = -\sum_i \sum_v p^{main}_i[v] \log p^{prx}_i[v] + \sum_i (\pi^{main}_{y_i \to y_{i+1}} - \pi^{prx}_{y_i \to y_{i+1}})^2$ where p^{main}_i and p^{prx}_i are the tag distributions computed by the main and proxy models respectively for the word $w_i \in \mathbf{w}$ (i.e., the scores $\mathbf{z}_i$). Note that p^{main}_i and $\pi^{main}_{y_i \to y_{i+1}}$ serve as the knowledge distillation signal that is obtained once the main model finishes its training at each iteration. Here, we will use the newly selected examples for the current annotation to compute the distillation signals. The overall objective to train M_{prx} at each AL iteration is thus: $L = L_{task} + L_{dist}$.

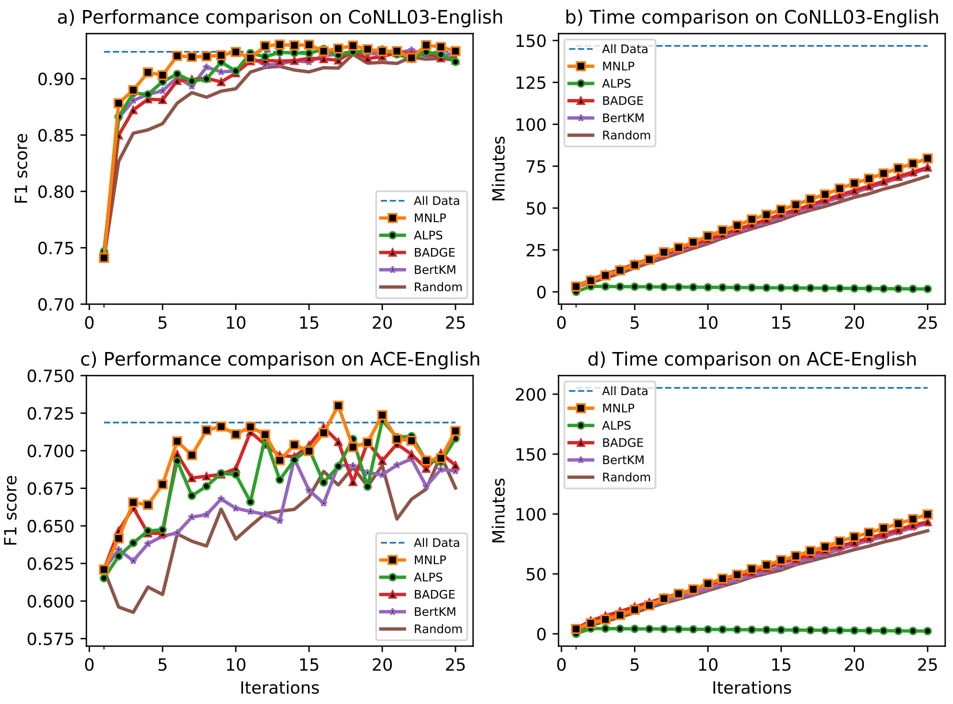


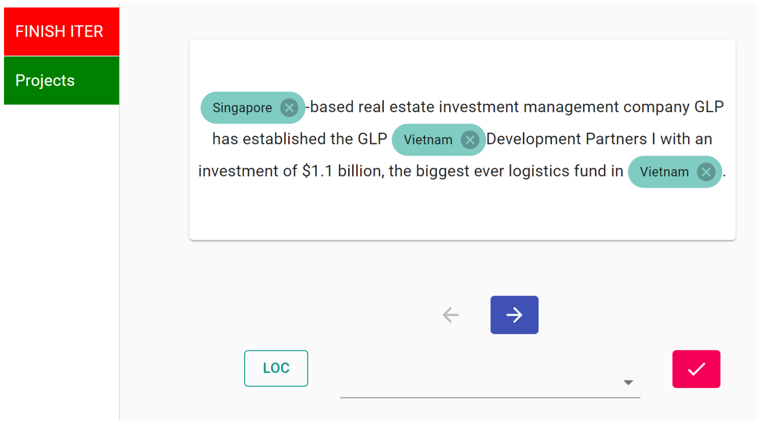
Figure 20. Comparison among data selection strategies.

4.2.3 Usage. Detailed documentation for FaMIE is provided at: `https://famie.readthedocs.io/`. The codebase is written in Python and Javascript, which can be easily installed through PyPI at : `https://pypi.org/project/famie/`.

Initialization. To initialize a project, users first choose a data selection strategy and upload a label set to define a sequence labeling problem. Next, the dataset U with unlabeled sentences should be submitted. FAMIE then allows users to interact with the models and annotate data over multiple rounds with a web interface. Also, FAMIE can detect languages automatically for further processing.

Annotating procedure. Given one annotation batch in an iteration, annotators label one sentence at a time as illustrated in Figure 21. In particular, the annotators annotate the word spans for each label by first choosing the label and then highlighting the appropriate spans. Also, FAMIE designs the size of the annotation batches to allow enough time to finish the training of the main model during the annotation time at each iteration.

Output. Unlike previous AL toolkits which focus only on their web interfaces to produce labeled data, FAMIE provides a simple and intuitive code interface for interacting with the resulting labeled dataset and trained main models after the AL processes. The code snippet in Figure 22 presents a minimal usage of our **famie** Python package to use the trained main model for inference over new data. This allows users to immediately evaluate their models and annotation efforts on data of interest.



Singapore -based real estate investment management company GLP has established the GLP Vietnam Development Partners I with an investment of $1.1 billion, the biggest ever logistics fund in Vietnam.

Figure 21. Annotation interface in FAMIE.

	Idle	CoNLL03-English						CoNLL02-Spanish						ACE-English						ACE-Chinese					
	mins/iter	10%	20%	30%	40%	50%	100%	10%	20%	30%	40%	50%	100%	10%	20%	30%	40%	50%	100%	10%	20%	30%	40%	50%	100%
Full Data	x	x	x	x	x	x	92.4	x	x	x	x	x	89.6	x	x	x	x	x	71.9	x	x	x	x	x	69.1
Large	41.6	**90.3**	**92.4**	**93.0**	**92.4**	92.4	x	86.9	**88.6**	**89.4**	**89.3**	89.0	x	**67.8**	**71.1**	**70.0**	**72.4**	**71.3**	x	**64.8**	67.6	**71.3**	68.7	**71.5**	x
FaMIE	**3.4**	90.1	91.7	91.8	91.7	**92.7**	x	86.5	88.2	88.5	88.1	**89.4**	x	67.0	69.3	69.5	68.9	70.6	x	61.3	**67.9**	68.5	**69.8**	69.6	x
FaMIE-A	5.7	89.7	90.8	91.3	91.9	91.7	x	**87.4**	87.2	89.0	87.7	89.1	x	67.2	68.0	69.5	68.9	70.6	x	62.8	66.5	67.9	66.3	69.4	x
FaMIE-AD	5.6	87.0	90.1	90.5	90.7	90.5	x	85.5	86.9	87.7	88.8	88.6	x	64.9	65.4	67.7	66.8	69.1	x	58.1	65.4	66.5	64.8	70.3	x
Random	x	86.0	89.1	90.6	91.4	91.9	x	80.8	85.3	88.1	88.7	88.6	x	60.4	64.1	66.9	69.0	67.5	x	48.4	58.2	65.1	65.4	66.6	x

Table 25. Main model's performance on multilingual NER and ED tasks. "Idle" indicate average waiting time of annotators.

```python
import famie
# access a project via its name
p = famie.get_project('NewProject')
# access the project's labeled data
data = p.get_labeled_data()

# access the project's trained target model
model = p.get_trained_model()
# make predictions with the trained model
doc = '''Nick is happy.'''
output = model.predict(doc)
print(output)
# [('Nick', 'B-Person'), ('is', 'O'), ('happy', 'O'), ('. ', 'O')]
```

Figure 22. Accessing the labeled dataset and the trained main model returned by an AL project.

4.2.4 Evaluation.

Datasets and Hyper-parameters. To comprehensively evaluate our AL framework FAMIE, we conduct experiments on two IE tasks (i.e., NER and ED) for three languages using four datasets: CoNLL03-English (Tjong Kim Sang & De Meulder, 2003) and CoNLL02-Spanish (Tjong Kim Sang, 2002) for NER, and ACE-English and ACE-Chinese for ED (i.e., using the multilingual ACE-05 dataset (T. H. Nguyen & Grishman, 2015a, 2018a; Walker et al., 2006)). The CoNLL datasets cover 4 entity types while 33 event types are annotated in ACE-05 datasets. We follow the standard data splits for train/dev/test portions for each dataset (V. D. Lai et al., 2020; Q. Li et al., 2013a; Pouran Ben Veyseh, Lai, et al., 2021).

For the main target model M_{main}, the full-scale XLM-Roberta$_{large}$ model is used as the encoder. Our framework for AL thus inherits the ability of XLM-Roberta to support more than 100 languages. Also, we employ the compact miniLM architecture (distilled from the pre-trained XLM-Roberta) for the proxy model M_{prx}. In all experiments, the main model is trained for 40 epochs while the proxy model is trained for 20 epochs at each iteration. We use the Adam optimizer with batch size of 16 and learning rate of 1e-5 to train the models.

We follow the AL settings in previous work to achieve consistent evaluation (Kim, 2020; M. Liu et al., 2022; Shelmanov et al., 2021). Specifically, the unlabeled pool is created by discarding labels from the original training data of each dataset; 2% of which ($\sim$ 242 sentences) is selected for labeling at each iteration for a total of 25 iterations (examples of the first iteration are randomly sampled to serve as the seed D_0). The annotation is simulated by recovering the ground-truth labels of the corresponding instances. The model performance is measured on the test datasets by taking average over 3 runs with different random seeds.

Comparing Data Selection Strategies. In this experiment, we aim to determine the best data selection strategy for our AL framework. To this end, we perform the standard AL process (i.e., training the full transformer-CRF model with no adapters, selecting data, and annotating data at each iteration) for different data selection strategies to measure performance and time. We focus on English datasets in this experiment. Figure 20 reports the performance across AL iterations of the model for different data selection methods. As can be seen, "*MNLP*" is the overall best method for data selection in AL. We will thus leverage MNLP as the data section strategy for the evaluation of FAMIE.

Also, Figure 20 shows the annotators' idle time (the combined time for model training and data selection) across iterations for each selection strategy. The major difference comes from ALPS that has significantly less waiting time than other methods as it does not require model training. However, ALPS's performance is considerably worse than MNLP as a result, especially in early iterations. This demonstrates the importance of training and including the main model during the AL iterations for data section. Importantly, we find that the waiting time of annotators at each iteration is very high in current AL methods (e.g., more than

30 minutes after the first 8 iterations with the MNLP strategy), thus affecting the annotators' productivity.

Performance and Time Efficiency. To evaluate the performance and time efficiency of FAMIE, Table 25 compares our full proposed framework FAMIE (with proxy model, knowledge distillation, and adapters) with the following baselines: (i) "**Large**": the best AL baseline from the previous experiment employing the full-scale transformer encoder and MNLP for data selection; (ii) "**Random**": this is the same as "**Large**", but replaces MNLP with random selection; (iii) "**FAMIE-A**": this is the proposed framework FAMIE without adapter-based tuning (all parameters from the main model are fine-tuned); and (iv) "**FAMIE-AD**": we further remove the knowledge distillation loss from "**FAMIE-A**" in this method. The experiments are done for all four datasets of NER and ED.

The first observation is that FAMIE's performance is only marginally below that of Large despite only using the small proxy network for data selection. Importantly, annotators only have to wait for about 3.4 minutes per AL iteration before they can annotate the next data batch in FAMIE. This is over 10 times faster compared to the standard AL approaches (e.g., in Large). Second, the adapters in FAMIE not only boost the overall performance for AL but also reduce the waiting time for annotators. Also, we note that using adapters, the training time of M_{main} only takes 32 minutes at each iteration (on average). This is reasonable to fit into the time that an annotator needs to spend to label an annotation batch at each AL iteration, thus accommodating our proposal for training the main model during annotation time. Finally, FAMIE-AD performs worst (i.e., similar or even worse than Random) in most cases, which confirms the necessity of our distillation component in FAMIE.

4.2.5 Related Work.

Despite the potential of AL in reducing annotation cost for a target task, most previous AL work focuses on developing data selection strategies to maximize the model performance (Ash et al., 2019; Kim, 2020; M. Liu et al., 2022; Margatina, Vernikos, Barrault, & Aletras, 2021; Sener & Savarese, 2017; D. Wang & Shang, 2014). As such, previous AL methods and frameworks tend to ignore the necessary time to train models and perform data selection at each AL iteration that can be significantly long and hinder annotators' productivity and model performance. To make AL frameworks practical, few recent works have attempted to minimize the model training and data selection time by leveraging simple and non state-of-the-art architectures as the main model, e.g., ActiveAnno (Wiechmann et al., 2021) and Paladin (Nghiem et al., 2021). However, an issue with these approaches is the inability to exploit recent advances in large-scale language models to achieve optimal performance. In addition, some recent works have also explored large-scale language models for AL (Shelmanov et al., 2021; M. Yuan et al., 2020); however, to reduce waiting time for annotators, such methods need to exclude the training of the large models in the AL iterations or employ small models for data selection, thus suffering from a harmful mismatch between the annotated examples and the main models (Lowell et al., 2019).

4.2.6 Summary.

We introduce FAMIE, a comprehensive AL framework that supports model creation and data annotation for sequence labeling in multiple languages. FAMIE optimizes the annotators' time by leveraging a small proxy network for data selection and a novel knowledge distillation to synchronize the proxy and main target models for AL. As FAMIE is task-agnostic, we plan to extend FAMIE to cover other NLP tasks in future work.

CHAPTER V
POTENTIAL APPLICATIONS OF INFORMATION EXTRACTION FOR ENHANCING LARGE LANGUAGE MODELS

This chapter contains materials from the published paper *"Minh Nguyen, Kishan K C, Toan Nguyen, Ankit Chadha, and Thuy Vu.* ***'Efficient fine-tuning large language models for knowledge-aware response planning'*** *In Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, 2023"* (M. Nguyen et al., 2023). Minh was responsible for the idea conception, model design, experiment setup, and writing as the first author. Kishan, Toan, Ankit, and Thuy provided meaningful discussions and analysis. Kishan and Thuy conducted the evaluation and contributed to the writing. The paper was revised to comply with the dissertation format and purposes.

The fourth and final research direction (RD4) explores the potential applications of information extraction (IE) for enhancing large language models (LLMs). This chapter introduces an innovative retrieval-augmented generation (RAG) framework called KARP as a case study, which comprises a novel knowledge retrieval component and an LLM for open-domain question answering. KARP employs IE techniques to convert unstructured text into structured data, facilitating the development of sophisticated retrieval systems that benefit RAG-based LLMs. The knowledge retriever in KARP extracts relevant words from web contexts to assess their relevance and determine the most suitable contexts for answer generation. We also propose a novel fine-tuning method for training the LLM to efficiently utilize both kinds of knowledge: external knowledge from web contexts, and internal knowledge embedded within the model parameters.

Experimental results demonstrate that KARP can provide natural, concise, and highly accurate answers for open-domain questions by leveraging the power of LLMs and the retrieval of relevant external knowledge, highlighting the potential of IE in enhancing LLMs for more effective and reliable language understanding and generation. This chapter serves as a starting point for discussing the broader potential of IE in improving various aspects of LLMs, such as knowledge retrieval, contextual understanding, and response generation, paving the way for future research and applications in this area.

5.1 Introduction

General question answering (QA), a crucial natural language processing (NLP) task, is often regarded as AI-complete (Clark et al., 2016; Weston et al., 2015); that is, QA will only be considered solved once all the challenging problems in artificial intelligence (AI) have been addressed. Several virtual response assistants, including Google Assistant, Amazon Alexa, and Apple's Siri, have integrated state-of-the-art QA technologies, allowing them to understand and generate responses in natural languages, providing valuable services to users. However, general QA still presents significant challenges, primarily due to the inherent difficulties in reasoning with natural language, including aspects like commonsense and general knowledge. Past research has explored the use of Large Language Models (LLMs) for general QA, predominantly leveraging either parametric (e.g., ChatGPT[1]) or external (e.g., WebGPT(Nakano et al., 2021)) knowledge sources. This method, however, can lead to considerable complications, including hallucination - the generation of plausible but incorrect or unverified information. To address these challenges, this paper introduces the concept of

[1]https://chat.openai.com/chat

Knowledge-Aware Response Planning (KARP) for general QA along with a novel framework that combines a knowledge retriever with a robust fine-tuning strategy for LLMs. In particular, the problem of KARP can be defined as follows. Given a user query and a prompt containing external knowledge, the goal is to develop a model that can consolidate a response that must be crafted not just from the externally sourced information, but also from the model's inherent parametric knowledge. This is different from the previous work that aim to generate a response by either harnessing parametric knowledge (e.g., ChatGPT) or retrieving from external knowledge such as knowledge bases (Bao, Duan, Yan, Zhou, & Zhao, 2016; Bao, Duan, Zhou, & Zhao, 2014; Saxena, Chakrabarti, & Talukdar, 2021; J. Xu et al., 2019), web documents (D. Chen, Fisch, Weston, & Bordes, 2017; D. Chen & Yih, 2020; Garg et al., 2020; W. Yang et al., 2019; Y. Yang, Yih, & Meek, 2015a), or a provided context (Devlin et al., 2019b; Hermann et al., 2015; Rajpurkar, Zhang, Lopyrev, & Liang, 2016; W. Wang, Yang, Wei, Chang, & Zhou, 2017).

With the emergent abilities of LLMs (Wei et al., 2022), generative QA systems, in which answers are produced by a generative LLM, have been explored to improve the performance of QA (Gabburo, Koncel-Kedziorski, Garg, Soldaini, & Moschitti, 2022; C.-C. Hsu, Lind, Soldaini, & Moschitti, 2021a; Izacard & Grave, 2021; Jiang, Araki, Ding, & Neubig, 2022; Lewis & Fan, 2019; Muller, Soldaini, Koncel-Kedziorski, Lind, & Moschitti, 2022; Raffel et al., 2020c; Roberts, Raffel, & Shazeer, 2020b). In paritcular, previous work typically employs pre-trained LLMs with encoder-decoder architectures such as BART (Lewis et al., 2020) and T5 (Raffel et al., 2020b), where the encoder consumes a given question and a *required* relevant context as input for the decoder to generate an answer to the question (C.-

q:	What college offers chiropractic ?
c_1:	New York Chiropractic College offers 1 Chiropractic Degree program. It's a private university in a far away town. In 2015, 173 students graduated in the study area of Chiropractic with students earning 173 Doctoral degrees.
a_1:	New York Chiropractic college offers chiropractic.
c_2:	Chiropractic care is also essential for college students who want to stay healthy. The central nervous system is based in the spinal column, so correcting subluxations (misalignments) of the spine is important, no matter how old you are. Holt Chiropractic in Port Orchard, WA provides expert chiropractic care to students of all ages.
a_2:	Holt Chiropractic College offers chiropractic.
c_3:	Howell Township is a township in Monmouth County, New Jersey, United States. As of the 2010 United States Census, the township's population was 51,075, reflecting an increase of 2,172 from the 48,903 counted in the 2000 Census.
a_3:	Howell Township College offers chiropractic.

Table 26. Generated answers for a question q with different context passages c_1 (relevant), c_2 (quasi-relevant), and c_3 (irrelevant) from MS MARCO QA NLG test set (T. Nguyen et al., 2016). Answers a_1, a_2, and a_3 are generated by GenQA (C.-C. Hsu et al., 2021b).

C. Hsu et al., 2021b; Khashabi et al., 2020). On one hand, the similarity between generative QA and the pre-training tasks of LLMs enables transfer learning to improve QA performance. On the other hand, the generative formulation allows for flexibility in handling various types of QA problems (e.g., extractive QA, multiple-choice QA) (Khashabi et al., 2020). However, a well-known issue that has been shown to occur with the generative models is hallucination (Maynez et al., 2020; Roller et al., 2021; Shuster, Poff, Chen, Kiela, & Weston, 2021a), where the models generate statements that are plausible looking but *factually* incorrect. Additionally, if the answers are composed by a pretrained LLM without external knowledge, i.e., using parametric knowledge, the information contained in the answers might be outdated and no longer valid. For example, the answer for the question *"Which country is the reigning World Cup champion?"* will change through time.

Recent works (C.-C. Hsu et al., 2021b; Nakano et al., 2021) mitigate these issues by employing an information retrieval component, which is responsible for collecting web content to compose an answer for a given question. Formally, given a question q and a retrieved web content c, the model is trained to take (q, c) as input to produce a response $a = f_\theta(q, c)$, where f_θ denotes the corresponding LLM with the parameters θ. Unfortunately, f_θ may merely learn to copy/synthesize information from c to produce a if c often contains necessary information for correctly answering the question q in training data. As a result, the model may fail to provide a correct answer for a given question if the retrieved content is missing or contains irrelevant information (see Table 26). In other words, performance of these retrieval-based QA models are limited to an upper bound by the knowledge retriever.

In this work, we address such issues in building a generative QA model. First, we utilize a knowledge retriever that employs Optimal Transport to extract relevant content from web documents or databases for a given user query. Second, we propose a novel fine-tuning strategy combining external knowledge, i.e., provided by the knowledge retriever and the intrinsic pre-trained knowledge in LLMs to generate informed responses. Particularly, we propose a novel knowledge retriever as answer reranking model. Our proposed model performs an alignment between a given question and a text passage via Optimal Transport to extract relevant words in web context for determining its correctness. The relevant words in the context will then be used to produce a correctness score for ranking. In this way, we can obtain top K relevant contexts from databases/web documents, which are treated as external knowledge in our framework. Different from the previous work that follows a single-stage finetuning strategy, we propose to employ a two-

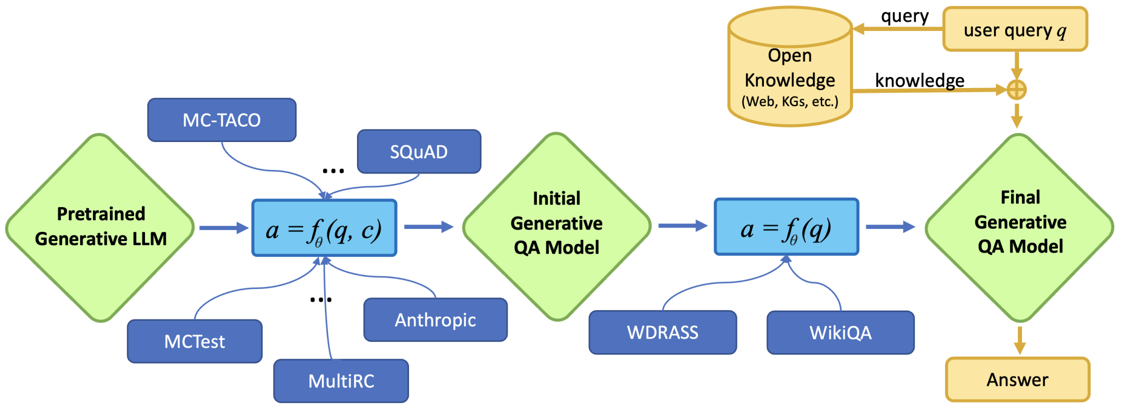


Figure 23. Overview of our proposed framework for KARP. The blue and orange arrows represent the finetuning and inference processes of our model respectively.

stage finetuning strategy, where both "$a = f_\theta(q, c)$" and "$a = f_\theta(q)$" templates are used to train the model. The latter intentionally excludes the external knowledge c from the input to encourage the model to exploit its own knowledge from the model parameters θ, which have been pretrained on massive unlabeled text (FitzGerald et al., 2022; Lewis et al., 2020; Raffel et al., 2020b; Soltan et al., 2022). To combine the two finetuning stages, we propose to finetune the LLM with the "$a = f_\theta(q, c)$" template, and sequentially finetune the model with "$a = f_\theta(q)$". At test time, we use the "$a = f_\theta(q, c)$" template to make predictions, where the context c is provided by our proposed knowledge retriever. Experimental results show that our proposed framework significantly improves the performance compared to the baselines on MS MARCO QA NLG (T. Nguyen et al., 2016), demonstrating the effectiveness of our proposed method. In addition, we also show that our proposed knowledge retriever contributes significantly to the overall performance of the system.

5.2 Proposed Method

Our proposed framework - KARP consists of (i) a knowledge retriever and (ii) a generative LLM-based answer generator. An overview of our framework is shown in Figure 23. Details regarding the knowledge retriever and the answer generator are presented in section 5.2.1 and 5.2.2, respectively.

5.2.1 Knowledge Retriever. Our knowledge retriever functions as an answer reranking model. Given a question q and a group of N web contexts $C = \{c_1, c_2, \ldots, c_N\}$, the goal is to determine the contexts containing the correct answer $A \subset C$ by learning a reranking function $r : Q \times \phi(C) \rightarrow \phi(C)$, where Q represents the set of questions and $\phi(C)$ represents all the possible orderings of C. The intent is to place the relevant contexts A at the top of the ranking produced by the function r. The reranker r is typically a pointwise network $f(q, c_i)$, such as TANDA (Garg et al., 2020), which learns to assign a relevance/correctness score $p_i \in (0, 1)$ to each c_i for ranking purposes.

Our knowledge retriever consists of three primary components: i) Encoding, ii) Question-Context Alignment, and iii) Answer-Context Dependencies. Overview of our proposed model is provided in Figure 24.

5.2.1.1 Encoding. We are provided with a question represented as $q = [w_1^q, w_2^q, \ldots, w_{T_q}^q]$ with T_q words and a set of N web contexts $C = \{c_1, c_2, \ldots, c_N\}$ retrieved from a search engine. Each context, denoted as $c_i = [w_1^c, w_2^c, \ldots, w_{T_c}^c]$, consists of T_c words. In this work, we consider previous and next sentences c_{prev}, c_{next} as additional contexts for each context $c \in C$. To create the input for our model, we concatenate the question, the web context, and context sentences into a single input sequence: $[q; c; c_{prev}; c_{next}]$. This combined sequence is then passed through a pre-trained language model (PLM), e.g., RoBERTa (Y. Liu et al., 2019), to obtain contextualized word embeddings. Additionally, we employ distinct segment embeddings for the question, the web context, and context sentences. These segment embeddings, which are randomly initialized and trainable during training, are added to the initial word embeddings in the first layer of the PLM. For simplicity, let $[\mathbf{w}_1^q, \mathbf{w}_2^q, \ldots, \mathbf{w}_{T_q}^q]$ and $[\mathbf{w}_1^c, \mathbf{w}_2^c, \ldots, \mathbf{w}_{T_c}^c]$ represent the sequences of word representations obtained from the last layer of the PLM for the question q and the web context $c \in C$, respectively.

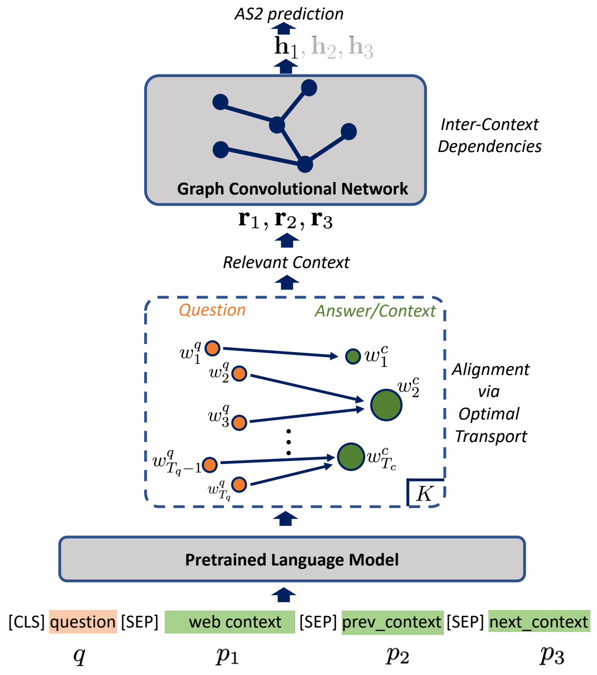

AS2 prediction
$\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3$
Graph Convolutional Network
Inter-Context Dependencies
$\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$
Relevant Context
Question
Answer/Context
w_1^q
w_2^q
w_3^q
$w_{T_q-1}^q$
$w_{T_q}^q$
w_1^c
w_2^c
$w_{T_c}^c$
K
Alignment via Optimal Transport
Pretrained Language Model
[CLS] question [SEP] web context [SEP] prev_context [SEP] next_context
q
p_1
p_2
p_3

Figure 24. A diagram depicting the knowledge retriever in our framework for KARP.

5.2.1.2 Question-Context Alignment. In this section, we present our approach for extracting relevant words within the web context and its surrounding sentences based on the alignment of words with the question. Specifically, we introduce the use of Optimal Transport (OT) (Cuturi, 2013; Monge, 1781) to address the task of aligning the question with the context for answer reranking.

OT is a well-established technique used to transfer probability from one distribution to another by establishing an alignment between two sets of points. In the discrete setting, we are provided with two probability distributions, denoted as

p_X and p_Y, defined over two sets of points, namely $X = \{x_i\}_{i=1}^{n}$ and $Y = \{y_j\}_{j=1}^{m}$ ($\sum_i p_{x_i} = 1$ and $\sum_j p_{y_j} = 1$). Additionally, a distance function $D(x, y) : X \times Y \to \mathbb{R}^+$ is given to quantify the distance between any two points x and y. The objective of OT is to determine a mapping that transfers the probability mass from the points in $\{x_i\}_{i=1}^{n}$ to the points in $\{y_j\}_{j=1}^{m}$, while minimizing the overall cost associated with this transportation. Formally, this involves finding the transportation matrix $\pi_{XY} \in \mathbb{R}^{+n\times m}$ that minimizes the following transportation cost:

$$d_{XY} = \sum_{\substack{1\le i\le n \\ 1\le j\le m}} D(x_i, y_j)\pi_{XY\,ij}, \tag{5.1}$$

so that $\pi_{XY}\mathbf{1}_m = p_X$ and $\pi_{XY}^T\mathbf{1}_n = p_Y$. The transportation matrix π_{XY} signifies the best matching between the sets of points X and Y, where each row i in the matrix indicates the optimal alignment from a point $x_i \in X$ to each point $y_j \in Y$.

In our problem of aligning the question with the web context, we treat the question q and the context c as two point sets: $\{w_i^q\}_{i=1}^{T_q}$ and $\{w_i^c\}_{i=1}^{T_c}$ respectively (each word is a point)[2]. To determine the probability distributions for these word sets, we propose calculating the word frequencies and then normalizing the sum of frequencies. Specifically, the probability distribution for the question is obtained by:

$$p_{w_i^q} = \frac{freq(w_i^q)}{\sum_{i'=1}^{T_q} freq(w_{i'}^q)} \tag{5.2}$$

The frequency $freq(w_i^q)$ corresponds to the number of occurrences of the word w_i^q in the training data. The same approach is applied to compute the probability distribution for the context. To handle unseen words during testing, we utilize Laplace smoothing to assign a non-zero probability. Moving on, we

[2]Before performing the alignment, we remove stopwords and punctuation marks from both sets of words.

estimate the distance between two words $w_i^q \in q$ and $w_j^c \in c$ by measuring their semantic divergence, which involves computing the Euclidean distance between their contextualized representations obtained from the PLM: $D(w_i^q, w_j^c) = ||\mathbf{w}_i^q - \mathbf{w}_j^c||$. The Sinkhorn-Knopp algorithm is then efficiently employed to solve for the optimal transportation matrix π_{XY} (in this case, π_{qc} for the question q and the context c) (Cuturi, 2013; Sinkhorn & Knopp, 1967). Finally, we obtain the relevant words r_c for the context c by taking the union of words w_j^c that have the highest transportation probabilities:

$$r_c = \bigcup_{i=1}^{T_q} \{w_j^c | j = \text{argmax}_{1 \leq j' \leq T_c} \pi_{qc_{ij'}}\} \tag{5.3}$$

To compute the representation for the context c, we take the average sum of the representations of the relevant words:

$$\mathbf{r}_c = \frac{1}{|r_c|} \sum_{j | w_j^c \in r_c} \mathbf{w}_j^c \tag{5.4}$$

By incorporating the information of the relevant words, our intention is to eliminate any disruptive or unrelated details from the web context.

5.2.1.3 Answer-Context Dependencies. For convenience, let $[\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$ denote the representations acquired from Equation (5.4) for the web context $p_1 \equiv c$, the previous context $p_2 \equiv c_{prev}$, and the next context $p_3 \equiv c_{next}$. To capture the relationships between these contexts, we view each context as a node in a fully-connected graph $G = (V, E)$, where $V = \{p_i\}$ $(1 \leq i \leq 3)$ is the node set and $E = \{(p_i, p_j)\}$ $(1 \leq i, j \leq 3)$ is the edge set. Our objective is to determine a weight $\alpha_{ij} \in (0, 1)$ for each edge (p_i, p_j) that reflects the dependency of p_i on p_j. To accomplish this, we propose to leverage their semantic representations $\mathbf{r}_i$, $\mathbf{r}_j$, and transportation costs to the question d_{qp_i}, d_{qp_j} to measure the dependency weight α_{ij} between the contexts p_i and p_j. Specifically, we first compute the score:

$u_{ij} = FFN_{DEP}([\mathbf{r}_i \odot \mathbf{r}_j; d_{qp_i}; d_{qp_j}])$, where $\odot$ is the element-wise product, $[;]$ represents the concatenation operation, and FFN_{DEP} is a feed-forward network. Subsequently, the weight α_{ij} for the edge (p_i, p_j) is obtained through a softmax function:

$$\alpha_{ij} = \frac{\exp(u_{ij})}{\sum_{j'=1}^{K} \exp(u_{ij'})} \tag{5.5}$$

The derived weights $\{\alpha_{ij}\}$ are subsequently utilized to enrich the passage representations through L layers of a Graph Convolutional Network (GCN) (Kipf & Welling, 2017):

$$\mathbf{h}_i^l = \mathrm{ReLU}(\sum_{j=1}^{K} \alpha_{ij} \mathbf{W}^l \mathbf{h}_j^{l-1} + \mathbf{b}^l) \tag{5.6}$$

where $\mathbf{W}^l$, $\mathbf{b}^l$ are learnable weight matrix and bias for the layer l of the GCN $(1 \le l \le L)$, and $\mathbf{h}_i^0 \equiv \mathbf{r}_i$ is the input representation for the context p_i. The output vectors $\mathbf{h}_i^L \equiv \mathbf{h}_i$ at the last layer of the GCN serve as the final representations for the context p_i. Intuitively, the weights α_{ij} enable each context to decide the amount of information it receives from the other contexts to improve its representation for the task. The representation $\mathbf{h}_1$ for the web context $p_1 \equiv c$ is finally sent to a feed-forward network with a sigmoid output function to estimate the relevance/correctness score $p_c \in (0, 1)$ for the context c: $p_c = FFN_{DPR}(\mathbf{h}_1)$. For training, we minimize the binary cross-entropy loss with the correctness scores p_c. At inference time, consistent with previous research (Garg et al., 2020), we include all web contexts for each question for ranking.

5.2.2 LLM-based Answer Generator.

5.2.2.1 Background on Text Generation Finetuning. Text generation finetuning has become a general approach to solving different NLP tasks, where input and expected output of a task can be represented as source and target text respectively for a generative model to learn the task (B. Lin et al., 2022; Lu et al., 2021b; Raffel et al., 2020b). For example, a pretrained generative LLM such as BART (Lewis et al., 2020) and T5 (Raffel et al., 2020b) can be finetuned on sentiment analysis by taking a statement (e.g., *"I really like the story"*) as the source text to generate a label (i.e., *"Positive"*, *"Negative"*, *"Neutral"*) to indicate the sentiment of the statement. As the text generation resembles the LLM's pretraining task (e.g., next word prediction), the formulation could facilitate the transfer learning to the target task. In addition, it enables data augmentation methods where training data for a task may also be leveraged for another task in the same generative formulation (J. Liu, Chen, Liu, Bi, & Liu, 2020). These advantages have led to significant performance improvements for many NLP tasks such as event extraction (J. Liu et al., 2020), named entity recognition (Yan et al., 2021), and dependency parsing (B. Lin et al., 2022). Similar to other NLP tasks, the generative methods have been explored for improving QA performance (Gabburo et al., 2022; C.-C. Hsu et al., 2021a; Izacard & Grave, 2021; Jiang et al., 2022; Lewis & Fan, 2019; Muller et al., 2022; Raffel et al., 2020c; Roberts et al., 2020b). To avoid hallucination and improve factual accuracy for the models, recent works on ODQA employ the retrieval-based methods such as GenQA (C.-C. Hsu et al., 2021b).

GenQA is introduced by Hsu et al. (C.-C. Hsu et al., 2021b) for generating appropriate answers for user questions given answer candidates retrieved by a reranking model. This expands the answer retrieval pipeline with an additional generative stage to produce correct and satisfactory answers, especially in cases where a highly ranked candidate is not acceptable or does not provide a natural response to the question. In particular, GenQA employs a pretrained generative LLM to produce an answer by taking a given question and a list of answer candidates as input, sorted by a trained reranking model.

5.2.2.2 Our Proposed Finetuning Method. The main goal of a general text-generation model is to produce an output text sequence $\mathbf{y} = [y_1, y_2, \ldots, y_T]$ based on a given input text sequence $\mathbf{x} = [x_1, x_2, \ldots, x_S]$, where the lengths of the input and output sequences are denoted by S and T, respectively. With a pretrained encoder-decoder LLM such as BART (Lewis et al., 2020) or T5 (Raffel et al., 2020b), we can compute the conditional probability of $P(\mathbf{y}|\mathbf{x})$ for training the model. At test time, the decoder merges the previous output and input text to create the current output. A decoding algorithm such as Greedy or Beam Search (Wiseman & Rush, 2016) can be used to generate an output text with the highest likelihood. For ODQA, given a question q and a retrieved web content c (e.g., top relevant contexts), previous works such as GenQA are trained to take (q, c) for as the source sequence to produce a response as the target sequence $a = f_\theta(q, c)$, where f_θ denotes the corresponding LLM with the parameters θ. As a result, f_θ may merely learn to copy/synthesize information from c to produce a if c often contains necessary information for correctly answering the question q in training data. Relying solely on the retrieved content c, the model may fail to provide a correct answer for a given question if c is missing or contains irrelevant/noisy information. In other words, performance of these retrieval-based QA models tend to be limited by an upper bound of the knowledge retriever's performance.

Different from the previous works that follow a single-stage finetuning method, we propose to employ a multi-stage finetuning method, where both "$a = f_\theta(q, c)$" and "$a = f_\theta(q)$" templates are used to train the model. The latter intentionally excludes the external knowledge c from the input to encourage the model to retrieve its own knowledge from the model parameters θ, which have been pretrained on massive unlabeled text (FitzGerald et al., 2022; Lewis et al., 2020; Raffel et al., 2020b; Soltan et al., 2022). To combine the two finetuning stages, we propose to finetune the LLM with "$a = f_\theta(q, c)$", and sequentially finetune the model with "$a = f_\theta(q)$". In this way, our model does not completely rely on the retrieval results to generate answers for given questions. At test time, we use the "$a = f_\theta(q, c)$" template to make predictions. The retrieved content c now can be considered as a source of external knowledge along with the pretrained knowledge contained in the model parameters θ to generate an answer for the question. Under this perspective, we consider various QA datasets for each step in our finetuning process. We call such dataset collection OKQA as they are publicly available and contains high-quality general knowledge.

MS Marco QA NLG is a specialized version of the MS Marco dataset (T. Nguyen et al., 2016) that aims to produce natural language responses to user inquiries using web search result excerpts. This dataset includes $182K$ queries from Bing search logs, each is associated with top ten most relevant passages. A human annotator is then required to look at the passages and synthesize an answer using the content of the passages that most accurately addresses the query.

Super Natural Instructions (SNI) is a data collection proposed by (Y. Wang et al., 2022). The corpus consists of $1,616$ diverse NLP tasks and their expert-written instructions. In this work, we consider only question-answering tasks such as extractive QA with SQUAD (Rajpurkar et al., 2016) and multiple-choice QA with MCTest (Richardson, Burges, & Renshaw, 2013). For each task, we consider anything but a question q provided in the input as context c. Particularly, the context c can be a passage, a fact, or a set of answer choices associated with the question. As a result, we obtain $180K$ examples for finetuning our model.

Anthropic is introduced by (Bai et al., 2022), containing conversations between a human and a computer assistant. For each conversation, we consider a human question in the current turn and the (question, answer) pairs in the previous turns as the input sequence. The answer from the assistant in the current turn is treated as the output sequence. In this way, the previous turns can be considered as a form of relevant context c for clarifying the current question q. Consequently, we obtain $280K$ examples for finetuning our model.

Answer Reranking datasets, namely, WikiQA (Y. Yang, Yih, & Meek, 2015b) and WDRASS (Zhang, Vu, Gandhi, Chadha, & Moschitti, 2022a) are also used for finetuning our model. WikiQA is a collection of questions and answer candidates that have been manually annotated using Bing query logs on Wikipedia. WDRASS is a large-scale dataset of questions that are non-factoid in nature, such as questions that begin with "why" and "how". The dataset contains around $64,000$ questions and over $800,000$ labeled passages that have been extracted from a total of $30M$ documents. Each question in such datasets is associated with a set of answer candidates, in which some of the candidates are correct answers. As a question can have multiple correct answers, we select the longest answer as the output sequence for the question, which is considered as the input sequence. This results in a set of $105K$ examples for finetuning our model.

In the end, the datasets where context is available for a question are used in the first stage of our finetuning process, while the other datasets are used for further training the model in the subsequent stage. With a huge amount of various QA tasks, we expect this could teach the model to understand the nature of question answering and how to utilize its own parametric knowledge (in case no context is provided) and external knowledge (i.e., relevant context) to answer a given question.

5.3 Experiments

5.3.1 Benchmarking the Knowledge Retriever.

5.3.1.1 Experimental Setup.

Datasets We follow the previous work (Garg et al., 2020; Zhang, Vu, Gandhi, Chadha, & Moschitti, 2022b) to conduct the evaluation. In particular, we use (i) **WikiQA** (Y. Yang et al., 2015b), consisting of questions from Bing query logs and manually annotated answers from Wikipedia, and (ii) **WDRASS** (Zhang et al., 2022b), a large-scale web-based dataset having factoid and non-factoid questions, to investigate our retrieval performance. We use the same train/dev/test splits used in previous work.

Hyper-parameters and Tools In accordance with previous work, we use a small portion of the WikiQA training data to tune hyper-parameters for our model and select the best hyper-parameters for all the datasets (Lauriola & Moschitti, 2021). We employ Adam optimizer to train the model with a learning rate of 1e-5 and a batch size of 64. We set 400 for the hidden vector sizes for all the feed-forward networks, $L = 2$ for the number of the GCN layers. We use Pytorch version 1.7.1 and Huggingface Transformers version 3.5.1 To implement the models. We use

Model	WikiQA				WDRASS	
	w/o ASNQ		with ASNQ		with ASNQ	
	P@1	MAP	P@1	MAP	P@1	MAP
TANDA	63.24*	75.00*	78.67*	86.74*	54.60	63.50
Ours	**74.16**	**83.29**	**83.77**	**89.28**	**55.9**	61.8

Table 27. Performance comparison on WikiQA and WDRASS, * indicates results reported by (Lauriola & Moschitti, 2021).

the NLTK library version 3.5 (Bird, Klein, & Loper, 2009) to preprocess the data and remove stopwords. The model performance is obtained over three runs with random seeds.

Evaluation Metrics We measure the model performance using the following standard metrics: Precision-at-1 (P@1) and Mean Average Precision (MAP) on the entire set of answer candidates for each question.

5.3.1.2 Performance Comparison. We compare our proposed model with TANDA (Garg et al., 2020), which is the current state-of-the-art model for answer reranking. Table 27 shows the performance comparison between the models in two settings: i) using a non-finetuned RoBERTa-Base encoder, and ii) using a fine-tuned RoBERTa-Base encoder. The non-finetuned RoBERTa-Base is obtained from (Y. Liu et al., 2019) while the other is produced by fine-tuning TANDA on the ASNQ dataset (Garg et al., 2020). As can be seen from the table, all the models benefit from using the finetuned RoBERTa-Base encoder. Across the two settings, our model outperforms the previous models by large margins, demonstrating its effectiveness for the task.

In addition, we show the performance of our proposed model compared to TANDA on the WDRASS test set. As we can see, our knowledge retriever significantly improves the performance for P@1 score, however, decreases the performance for MAP score. We attribute this to the fact that questions in WDRASS dataset usually have more than 1 correct answers for a single question while our model ranks the answer candidates individually. However, we note that the top-1 answer candidate is often the most helpful for the answering process.

5.3.1.3 Ablation Study. To understand the impact of each component in our proposed model, we conduct ablation experiments by removing/replacing different components in our model and evaluating the ablated models on the WikiQA development data.

Impact of Individual Components: First, we exclude each component from our proposed model to obtain the ablated models: *"- OT alignment"* (removing the question-candidate alignment via Optimal Transport), and *"- GCN dependencies"* (removing the inter-candidate dependencies via GCN). As shown in the Table 28, the removal of each component results in significant drops in the performance of the models, demonstrating the contributions of each component to the overall performance of our model.

Models	**P@1**	**MAP**	**MRR**
TANDA	81.2	88.6	88.9
Our Model	85.3	89.9	90.6
− *OT alignment*	83.6	89.1	89.6
− *GCN dependencies*	84.4	88.7	89.3

Table 28. Performance of ablated models on WikiQA development data for each component in our proposed answer reranking model.

Designs for Question-Candidate Alignment: Second, we experimented with different design choices for our question-candidate alignment component. Specifically, we tried the following models: *"+uniform dist"* (replacing the frequency-based distributions for OT with uniform distributions), and *"+cosine distance"* (employing the cosine distance instead of the Euclidean distance for

OT). As shown in Table 29, the performance of the ablated models decreases. This validates our design choices for the question-candidate alignment via OT. Additionally, we incorporated the question-candidate alignment into the TANDA baseline, where the alignment happens between a question and an answer candidate. The resulting model obtains significant improvement, showing the effectiveness of the question-candidate alignment for the task.

Models	**P@1**	**MAP**	**MRR**
Our Model	85.3	89.9	90.6
+ *uniform dist*	84.4	89.5	90.1
+ *cosine distance*	83.6	89.4	89.9
− *OT* + *cosine*	83.6	89.0	89.5
TANDA	81.2	88.6	88.9
+ *OT alignment*	83.6	89.3	89.6

Table 29. Performance of ablated models on WikiQA development data for the question-candidate alignment.

Learning Inter-Context Dependencies: Third, we would like to understand the effects of the following ablated models in capturing the dependencies among the contexts: *"- transportation costs"* (removing the OT transportation costs d_{qc_i} and d_{qc_j} from the computations of the dependency weights), *"+ vector concatenation"* (concatenating the candidate representations $\mathbf{r}_i$ and $\mathbf{r}_j$ instead of element-wise multiplying them), and *"+ cosine weights"* (computing dependency weights α_{ij} via the cosine similarity between the representations $\mathbf{r}_i$, $\mathbf{r}_j$ for the answer contexts). The incline of the ablated models' performance in Table 30 confirms the effectiveness of our proposed method for learning the dependencies among the answer contexts.

5.3.2 Automatic Evaluation for Knowledge-Aware Answer Planning.

Models	**P@1**	**MAP**	**MRR**
Our Model	85.3	89.9	90.6
− *transportation costs*	83.6	89.2	89.8
+ *vector concatenation*	84.4	89.5	90.3
+ *cosine weights*	83.6	88.7	89.0

Table 30. Performance of ablated models on WikiQA development data for the inter-candidate dependencies.

5.3.2.1 Experimental Setup. **Dataset**: We acquire the evaluation data as follows. First, we randomly select 2,000 questions from the MS MARCO QA NLG test set. For each question, we rank all the retrieval contexts using our proposed reranking model trained on WDRASS to obtain the top 5 candidates. We then concatenate the question and contexts to form the input, which is used to generate the predicted answer.

Hyper-parameters and Tools: To train the answer generators, we employ the Adam optimizer with a learning rate of 1e-5 and a batch size of 128. The implementation of the models is carried out using Pytorch version 1.7.1 and Huggingface Transformers version 3.5.1. Unless otherwise specified, all the models employ the pretrained T5-large as the base model.

Evaluation Metrics: We employ widely-used evaluation metrics, including ROUGE (C.-Y. Lin, 2004), BLEU (Papineni, Roukos, Ward, & Zhu, 2002), and BERTScore (Zhang*, Kishore*, Wu*, Weinberger, & Artzi, 2020), for assessing the quality of generated answers in comparison to human-written natural answers. These metrics are commonly applied to standard text generation tasks such as summarization (Zhang, Zhao, Saleh, & Liu, 2020), machine translation (Vaswani et al., 2017), and answer generation (Raffel et al., 2020c).

It is important to note that these metrics have their own limitations; however, these can be mitigated by providing more and higher-quality reference

Model	BLEU	RougeL	BERTScore
GenQA (C.-C. Hsu et al., 2021b)	14.6	0.518	0.698
KARP (Ours)	38.3	0.632	0.762

Table 31. Comparison between KARP and GenQA (C.-C. Hsu et al., 2021b) using automatic evaluation metrics.

texts (Callison-Burch, Osborne, & Koehn, 2006). In the context of answer generation, we enhance the reliability of these measurements by employing human-written answers as references.

5.3.2.2 Performance Comparison. Table 31 presents a comparison of KARP with GenQA in terms of BLEU, RougeL, and BERTScore metrics.

The results demonstrate that KARP outperforms GenQA in all evaluation metrics. KARP achieves a BLEU score of 39.4, a RougeL score of 0.608, and a BERTScore of 0.752. These results indicate that KARP offers a significant improvement over GenQA in the context of answer generation, which we attribute to our specialized fine-tuning method.

5.3.3 Human Evaluation for Knowledge-Aware Response Planning.

In this section, we evaluate KARP in an end-to-end industry-scale scenario.

5.3.3.1 Experimental Setup. We outline the experimental setup to evaluate the end-to-end performance of KARP in a web-scale scenario, involving tens of millions of web documents. The configuration allows us to study the scalability and effectiveness of our approach in a real-world, large-scale setting.

Web Document Collection: We constructed a large collection of web data, comprising documents and passages, to facilitate the development of knowledge retrieval for end-to-end system evaluation. This resource enables us to assess the impact of our work in an industry-scale ODQA setting. We selected

English web documents from the top 5,000 domains, including Wikipedia, from Common Crawl's 2019 and 2020 releases. The pages were split into passages following the dense passage retrieval (DPR) procedure (Karpukhin et al., 2020), limiting passage length to 200 tokens while maintaining sentence boundaries. This produced a collection of roughly 100 million documents and 130 million passages. From this, we built (i) a standard Lucene/Elasticsearch index and (ii) a neural-based DPR index (Karpukhin et al., 2020).

Web-scale Knowledge Retrieval: For each question, we retrieved up to 1,000 documents/passages using both indexes. We then rank the passages and applied a knowledge retriever to select relevant contexts. We used top $K = 5$ contexts as external knowledge for a question.

Question Sampling: We randomly selected 2,000 questions from WDRASS test set as it shows to represent natural questions extracted from the Web. In addition, the questions were also manually labeled.

Baselines: We employ GenQA (C.-C. Hsu et al., 2021b) as our main baseline in this experiment.

Evaluation Metrics: We evaluate the performance of the end-to-end QA system using accuracy metrics, i.e., the percentage of questions that were answered satisfactorily, judged by human experts. Additionally, we define a correct answer as one that must not only be factually accurate, but also expressed in a natural and fluent manner. Answers that are too verbose or oddly phrased are considered unsatisfactory.

5.3.3.2 Performance Comparison. Table 32 presents the relative accuracy of different QA settings, including TANDA (Garg et al., 2020), GenQA (C.-C. Hsu et al., 2021b), and our proposed KARP. As we can see, using

Model	**Accuracy**
TANDA	*baseline*
TANDA → GenQA	+2.20%
TANDA → KARP	+4.50%
KARP → KARP	+6.20%
KARP → KARP (OKQA)	+7.40%

Table 32. Relative accuracy of different QA settings: TANDA (Garg et al., 2020), GenQA (C.-C. Hsu et al., 2021b), and our proposed frame work.

GenQA to generate an answer based on the answer candidates retrieved by TANDA helps improve the accuracy by +2.2% (TANDA → GenQA). The performance is then improved further by +4.5% when TANDA is coupled with the model finetuned using KARP for answer generation ("TANDA → GenQA"), which shows the clear benefit of our two-stage finetuning method compared to GenQA. If both our proposed knowledge retriever and finetuning technique are employed, the performance boost compared to TANDA achieves at +6.2% ("KARP → KARP"). This demonstrates the importance of our proposed knowledge retriever in providing better answer candidates for the answer generation of the model. Finally, the best performer among all the models is "KARP → KARP (OKQA)", achieved when we apply KARP with additional training data from OKQA to improve the performance of TANDA by +7.4%. The result further demonstrates the efficacy of our proposed method for open domain question answering.

5.4 Related Work

Large Language Models (LLMs): LLMs have transformed NLP technologies with the advent of the Transformer architecture (Vaswani et al., 2017). Two fundamental pre-training objectives, Masked Language Modeling (MLM) and Causal Language Modeling (CLM), underpin the success of these models. MLM, introduced by BERT (Devlin et al., 2019b), predicts masked tokens in a sentence using surrounding context, enabling LLMs to learn bidirectional representations that excel in various NLP tasks. In contrast, CLM, exemplified by GPT (Radford, Narasimhan, Salimans, & Sutskever, 2018), predicts the next token in a sequence given its preceding context, showing remarkable success in text generation and other downstream applications (Kaplan et al., 2020; Radford et al., 2019; Raffel et al., 2020c). In this paper, we leverage the CLM architecture for its language generation capabilities to enhance QA performance.

General Question Answering using LLM: A standard QA system consists of (i) a retrieval engine that returns relevant knowledge and (ii) a model that generates a response addressing the question, either through selection (Garg et al., 2020; Severyn & Moschitti, 2015; Yoon, Dernoncourt, Kim, Bui, & Jung, 2019) or abstractive summarization of the top-selected answers (Gabburo et al., 2022; C.-C. Hsu et al., 2021a; Muller et al., 2022). In particular, recent summarization-based approaches, e.g., GenQA (Gabburo et al., 2022; C.-C. Hsu et al., 2021a; Muller et al., 2022), are highly susceptible to hallucination due to the absence of special treatment of irrelevant candidates, which commonly appear among the top-ranked options. As a result, the generated answer may seem plausible but could be factually incorrect (Ji et al., 2023; Raunak, Menezes, & Junczys-Dowmunt, 2021; Rebuffel et al., 2021; Shuster, Poff, Chen, Kiela, & Weston, 2021b; C. Wang & Sennrich, 2020; Xiao & Wang, 2021; Zhao, Cohen, & Webber, 2020; C. Zhou et al., 2021b). Even though its original goal is to generate more natural answers, GenQA (Gabburo et al., 2022; C.-C. Hsu et al., 2021a; Muller et al., 2022) can be considered as a method to ground LLMs for QA as it decodes an answer from the concatenation of both question and answer candidates. This approach, however, requires good answer candidates and careful finetuning to reduce hallucinations.

We propose, instead, a novel generation-based approach that leverages the emerging language reasoning capabilities of Large Language Models (LLMs) (Radford et al., 2018) to enhance quality of generated answers. In particular, KARP is designed to mitigate the reliance on oracle data by making use of the context, such as all choices in multiple-choice QA, instead of a correct answer alone, i.e., the correct choice. The experiments demonstrated that our proposed framework for KARP is highly resilient to noisy input data, and bring about broader application across different QA tasks.

Fine-tuning Strategies for LLMs: Several fine-tuning strategies have been specifically proposed for large language models (LLMs). These strategies can be broadly categorized into two groups: architecture-centric and data-centric. (i) Architecture-centric fine-tuning aims to improve the model's robustness and adaptability by modifying hyper-parameters across layers. Gradual unfreezing (Howard & Ruder, 2018) is one example, involving sequential fine-tuning of model layers to prevent catastrophic forgetting and better adapt to downstream tasks. Layer-wise learning rate decay (Radford et al., 2018) is another example, where different learning rates are assigned to various layers to enable more refined adaptation to the target task. (ii) Data-centric fine-tuning, on the other hand, concentrates on leveraging data from different sources or intermediate tasks to enhance model performance. Sequential fine-tuning (Garg et al., 2020; Gururangan et al., 2020) involves training the model on intermediate tasks before the final target task, improving its performance on the latter. Combining several related datasets for multi-task fine-tuning has also been shown to improve performance on the target task (X. Liu, He, Chen, & Gao, 2019). Our work is related to data-centric fine-tuning. In particular, we propose a novel strategy specifically designed for the question answering context. By leveraging both external knowledge and intrinsic parametric knowledge of LLMs, our approach aims to enhance the quality of generated answers in QA tasks.

5.5 Summary

In this chapter, we introduced KARP, a novel Retrieval-Augmented Generation (RAG) framework for Open-Domain Question Answering (ODQA). KARP consists of a novel knowledge retriever and an LLM-based answer generation component. Our experimental results demonstrate that the proposed knowledge retriever can obtain significantly higher quality contexts compared to TANDA, the state-of-the-art reranking model for ODQA. This finding highlights the benefit of incorporating Information Extraction (IE) techniques in building advanced retrieval systems for Large Language Models (LLMs).

Furthermore, we proposed a two-stage finetuning method that outperforms GenQA, the standard fine-tuning approach for RAG-based LLMs, in various settings. This result underscores the importance of leveraging the intrinsic parametric knowledge of LLMs in addition to the retrieved contexts to enhance their performance in ODQA tasks. By effectively utilizing the LLMs' inherent knowledge, our approach achieves superior results compared to relying solely on the information provided by the retrieval contexts.

CHAPTER VI
CONCLUSIONS

I was the main author of this chapter and Thien Nguyen provided editorial suggestions.

6.1 Summary

This dissertation has undertaken a comprehensive exploration into the domain of Multilingual Information Extraction (Multilingual IE) within the broader field of Natural Language Processing (NLP). Through the dedication to understanding and enhancing upstream models, developing language-agnostic downstream architectures, and innovating cross-lingual transfer learning and active learning methods, significant strides have been made towards a more inclusive, equitable, and linguistically diverse digital future. Notably, the research has underscored the vital role of IE in the evolution and improvement of large language models (LLMs), especially through the introduction of a novel retrieval-augmented generation (RAG) framework. The culmination of this work presents a significant contribution to the field of NLP and Multilingual IE, aiming at bridging the global communication gap and ensuring information accessibility and cultural preservation across a myriad of languages.

6.2 Limitations

Despite the considerable progress and achievements, this dissertation acknowledges several limitations that warrant further discussion:

– Data Scarcity for Low-Resource Languages: While strides have been made in developing methods for IE in low-resource languages, the scarcity of digital resources and annotated datasets remains a significant challenge. The effectiveness of these methods can still be constrained by the availability and quality of data for training models.

– Complexity of Linguistic Diversity: The intrinsic complexity and variability of human language across different cultures and linguistic structures pose ongoing challenges to creating universally effective IE models. While the research has made advancements in language-agnostic architectures, capturing the full range of linguistic nuances remains an area for further enhancement.

– Model Generalizability and Scalability: While efforts have been directed towards developing scalable and generalizable models, ensuring these models' robustness across an extensive array of languages and contexts is an area that requires continuous refinement and testing.

6.3 Future Works

Looking ahead, the following avenues for future research emerge as critical steps towards overcoming the limitations identified and pushing the boundaries of Multilingual IE further:

– Enhanced Data Acquisition and Annotation for Low-Resource Languages: Innovative approaches to data generation, such as synthetic data creation or semi-supervised learning methods, could mitigate the impact of data scarcity. Additionally, collaborative global initiatives to annotate data in low-resource languages can significantly contribute to this effort.

– Deeper Exploration of Cross-Linguistic and Cultural Nuances: Future research should delve into more sophisticated models that can better understand and interpret the subtleties of cultural and linguistic diversity.

This includes models that can dynamically adapt to the context and cultural background of the text being processed.

– Further Development of RAG Frameworks for LLMs: Building upon the introduced RAG framework, future works could focus on enhancing the knowledge retrieval components to improve the accuracy and relevance of information sourced by LLMs. This would include the refinement of IE techniques to structure unstructured data more effectively, thereby improving the quality of inputs for LLMs.

In conclusion, while this dissertation has made substantial contributions to the field of Multilingual IE, the path forward invites a collaborative, innovative, and multifaceted research effort. By addressing the limitations and embracing the proposed future directions, the next generation of NLP research can continue to make significant advances towards a more connected, inclusive, and linguistically diverse digital world.

REFERENCES CITED

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., … others (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Adel, T., Zhao, H., & Wong, A. (2017). Unsupervised domain adaptation with a relaxed covariate shift assumption. In *Proceedings of the association for the advancement of artificial intelligence (aaai).*

Adelani, D. I., Abbott, J., Neubig, G., D'souza, D., Kreutzer, J., Lignos, C., … Osei, S. (2021). MasakhaNER: Named entity recognition for African languages. *Transactions of the Association for Computational Linguistics*, *9*, 1116–1131. Retrieved from `https://aclanthology.org/2021.tacl-1.66` doi: 10.1162/tacl_a_00416

Aharoni, R., Johnson, M., & Firat, O. (2019, June). Massively multilingual neural machine translation. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 3874–3884). Minneapolis, Minnesota: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N19-1388` doi: 10.18653/v1/N19-1388

Ahmad, W. U., Peng, N., & Chang, K.-W. (2021). Gate: Graph attention transformer encoder for cross-lingual relation and event extraction. In *Proceedings of the association for the advancement of artificial intelligence (aaai).*

Akbik, A., Bergmann, T., Blythe, D., Rasul, K., Schweter, S., & Vollgraf, R. (2019, June). FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics (demonstrations)* (pp. 54–59). Minneapolis, Minnesota: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N19-4010` doi: 10.18653/v1/N19-4010

Ash, J. T., Zhang, C., Krishnamurthy, A., Langford, J., & Agarwal, A. (2019). Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., … others (2022). Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Banerjee, S., & Ghosal, S. (2015). Bayesian structure learning in graphical models. *Journal of Multivariate Analysis*, *136*, 147–162.

Bao, J., Duan, N., Yan, Z., Zhou, M., & Zhao, T. (2016). Constraint-based question answering with knowledge graph. In *Proceedings of coling 2016, the 26th international conference on computational linguistics: technical papers* (pp. 2503–2514).

Bao, J., Duan, N., Zhou, M., & Zhao, T. (2014). Knowledge-based question answering as machine translation. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 967–976).

Bekoulis, G., Deleu, J., Demeester, T., & Develder, C. (2018a). Adversarial training for multi-context joint entity and relation extraction. In *Proceedings of the 2018 conference on empirical methods in natural language processing.*

Bekoulis, G., Deleu, J., Demeester, T., & Develder, C. (2018b, October-November). Adversarial training for multi-context joint entity and relation extraction. In *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 2830–2836). Brussels, Belgium: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D18-1307` doi: 10.18653/v1/D18-1307

Benikova, D., Biemann, C., & Reznicek, M. (2014, May). NoSta-D named entity annotation for German: Guidelines and dataset. In *Proceedings of the ninth international conference on language resources and evaluation (LREC'14)* (pp. 2524–2531). Reykjavik, Iceland: European Language Resources Association (ELRA). Retrieved from `http://www.lrec-conf.org/proceedings/lrec2014/pdf/276_Paper.pdf`

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Blommaert, J. (2013). Language and the study of diversity. *Tilburg Papers in Culture Studies*.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146. Retrieved from `https://aclanthology.org/Q17-1010` doi: 10.1162/tacl_a_00051

Borisov, O., Aliannejadi, M., & Crestani, F. (2021). Keyword extraction for improved document retrieval in conversational search. *CoRR*, *abs/2109.05979*. Retrieved from `https://arxiv.org/abs/2109.05979`

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., . . . others (2020). Language models are few-shot learners. *Advances in neural information processing systems*, *33*, 1877–1901.

Callison-Burch, C., Osborne, M., & Koehn, P. (2006, April). Re-evaluating the role of Bleu in machine translation research. In *11th conference of the European chapter of the association for computational linguistics* (pp. 249–256). Trento, Italy: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/E06-1032`

Cao, Y., Liu, H., & Wan, X. (2020, July). Jointly learning to align and summarize for neural cross-lingual summarization. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 6220–6231). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-main.554` doi: 10.18653/v1/2020.acl-main.554

Che, W., Feng, Y., Qin, L., & Liu, T. (2020). N-ltp: A open-source neural chinese language technology platform with pretrained models. *arXiv preprint arXiv:2009.11616*.

Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017). Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Chen, D., & Yih, W.-t. (2020). Open-domain question answering. In *Proceedings of the 58th annual meeting of the association for computational linguistics: tutorial abstracts* (pp. 34–37).

Chen, X., Awadallah, A. H., Hassan, H., Wang, W., & Cardie, C. (2019, July). Multi-source cross-lingual model transfer: Learning what to share. In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 3098–3112). Florence, Italy: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P19-1299` doi: 10.18653/v1/P19-1299

Chen, X., & Cardie, C. (2018, October-November). Unsupervised multilingual word embeddings. In *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 261–270). Brussels, Belgium: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D18-1024` doi: 10.18653/v1/D18-1024

Chen, Y., Xu, L., Liu, K., Zeng, D., & Zhao, J. (2015a). Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing.*

Chen, Y., Xu, L., Liu, K., Zeng, D., & Zhao, J. (2015b). Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the annual meeting of the association for computational linguistics (acl).*

Chen, Y., Xu, L., Liu, K., Zeng, D., & Zhao, J. (2015c, July). Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)* (pp. 167–176). Beijing, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P15-1017` doi: 10.3115/v1/P15-1017

Chiu, J. P., & Nichols, E. (2016). Named entity recognition with bidirectional lstm-cnns. In *Transactions of the association for computational linguistics.*

Chow, C., & Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory*, *14*(3), 462–467.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., ... others (2023). Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, *24*(240), 1–113.

Chu, Y.-J. (1965). On the shortest arborescence of a directed graph. *Scientia Sinica.*

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., ... others (2022). Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416.*

Cicek, S., & Soatto, S. (2019). Unsupervised domain adaptation via regularized conditional alignment. In *Proceedings of the international conference on computer vision (iccv).*

Clark, P., Etzioni, O., Khot, T., Sabharwal, A., Tafjord, O., Turney, P., & Khashabi, D. (2016, Mar.). Combining retrieval, statistics, and inference to answer elementary science questions. *Proceedings of the AAAI Conference on Artificial Intelligence*, *30*(1). doi: 10.1609/aaai.v30i1.10325

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., ... Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116.*

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., . . . Stoyanov, V. (2020, July). Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 8440–8451). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-main.747` doi: 10.18653/v1/2020.acl-main.747

Corcoglioniti, F., Dragoni, M., Rospocher, M., & Aprosio, A. P. (2016). Knowledge extraction for information retrieval. In *The semantic web. latest advances and new domains: 13th international conference, eswc 2016, heraklion, crete, greece, may 29–june 2, 2016, proceedings 13* (pp. 317–333).

Cuturi, M. (2013). Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, *26*.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019a). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: Human language technologies.*

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019b, June). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 4171–4186). Minneapolis, Minnesota: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N19-1423` doi: 10.18653/v1/N19-1423

de Vries, W., van Cranenburgh, A., Bisazza, A., Caselli, T., van Noord, G., & Nissim, M. (2019). Bertje: A dutch bert model. *arXiv preprint arXiv:1912.09582*.

Dozat, T., & Manning, C. D. (2017). Deep biaffine attention for neural dependency parsing. In *Proceedings of the international conference on learning representations.*

Du, X., & Cardie, C. (2020, November). Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 671–683). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.emnlp-main.49` doi: 10.18653/v1/2020.emnlp-main.49

Eaton, D., & Murphy, K. (2012). Bayesian structure learning using dynamic programming and mcmc. *arXiv preprint arXiv:1206.5247*.

Edmonds, J. (1967). Optimum branchings. *Journal of Research of the national Bureau of Standards B*.

Ekbal, A., Haque, R., & Bandyopadhyay, S. (2007). Bengali part of speech tagging using conditional random field. In *Proceedings of the seventh international symposium on natural language processing, snlp-2007.*

Evans, N. (2018). The dynamics of language diversity. In *The dynamics of language: Plenary and focus lectures from the 20th international congress of linguists* (pp. 12–41).

Feng, A., You, C., Wang, S., & Tassiulas, L. (2022). Kergnns: Interpretable graph neural networks with graph kernels. In *Proceedings of the aaai conference on artificial intelligence.*

FitzGerald, J. G. M., Ananthakrishnan, S., Arkoudas, K., Bernardi, D., Bhagia, A., Bovi, C. D., . . . Natarajan, P. (2022). Alexa teacher model: Pretraining and distilling multi-billion-parameter encoders for natural language understanding systems. In *Kdd 2022.*

Forney, G. D. (1973). The viterbi algorithm. *Proceedings of the IEEE*, *61*(3), 268–278.

Fu, L., Nguyen, T. H., Min, B., & Grishman, R. (2017, November). Domain adaptation for relation extraction with domain adversarial neural network. In *Proceedings of the eighth international joint conference on natural language processing (volume 2: Short papers)* (pp. 425–429). Taipei, Taiwan: Asian Federation of Natural Language Processing. Retrieved from `https://aclanthology.org/I17-2072`

Fu, T.-J., Li, P.-H., & Ma, W.-Y. (2019). GraphRel: Modeling text as relational graphs for joint entity and relation extraction. In *Proceedings of the 57th annual meeting of the association for computational linguistics.*

Gabburo, M., Koncel-Kedziorski, R., Garg, S., Soldaini, L., & Moschitti, A. (2022, December). Knowledge transfer from answer ranking to answer generation. In *Proceedings of the 2022 conference on empirical methods in natural language processing* (pp. 9481–9495). Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.

Ganin, Y., & Lempitsky, V. (2015). Unsupervised domain adaptation by backpropagation. In *International conference on machine learning* (pp. 1180–1189).

Gao, H., Pei, J., & Huang, H. (2019). Conditional random field enhanced graph convolutional neural networks. In *Proceedings of the 25th acm sigkdd international conference on knowledge discovery & data mining* (pp. 276–284).

Garg, S., Vu, T., & Moschitti, A. (2020, Apr). Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(05), 7780–7788. Retrieved from `http://dx.doi.org/10.1609/AAAI.V34I05.6282` doi: 10.1609/aaai.v34i05.6282

Gärtner, T., Flach, P., & Wrobel, S. (2003). On graph kernels: Hardness results and efficient alternatives. In *Learning theory and kernel machines* (pp. 129–143). Springer.

Giunchiglia, F., Batsuren, K., Bella, G., et al. (2017). Understanding and exploiting language diversity. In *Ijcai* (pp. 4009–4017).

Gumbel, E. J. (1948). Statistical theory of extreme values and some practical applications: a series of lectures. In *Us government printing office.*

Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., & Smith, N. A. (2020, July). Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 8342–8360). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-main.740` doi: 10.18653/v1/2020.acl-main.740

Gutmann, M. U., & Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, *13*(2).

Han, R., Ning, Q., & Peng, N. (2019). Joint event and temporal relation extraction with shared representations and structured prediction. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing.*

He, K., Yan, Y., & Xu, W. (2020). Adversarial cross-lingual transfer learning for slot tagging of low-resource languages. In *Proceedings of the international joint conference on neural networks (ijcnn).*

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., & Blunsom, P. (2015). Teaching machines to read and comprehend. *Advances in neural information processing systems*, *28*.

Heyman, G., Verreet, B., Vulić, I., & Moens, M.-F. (2019, June). Learning unsupervised multilingual word embeddings with incremental multilingual hubs. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 1890–1902). Minneapolis, Minnesota: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N19-1188` doi: 10.18653/v1/N19-1188

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., ... Gelly, S. (2019). Parameter-efficient transfer learning for nlp. In *Proceedings of the international conference on machine learning.*

Hovy, D., & Prabhumoye, S. (2021). Five sources of bias in natural language processing. *Language and linguistics compass*, *15*(8), e12432.

Howard, J., & Ruder, S. (2018, July). Universal language model fine-tuning for text classification. In *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 328–339). Melbourne, Australia: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P18-1031` doi: 10.18653/v1/P18-1031

Hsu, C.-C., Lind, E., Soldaini, L., & Moschitti, A. (2021a). Answer generation for retrieval-based question answering systems. In *Acl findings 2021.*

Hsu, C.-C., Lind, E., Soldaini, L., & Moschitti, A. (2021b, August). Answer generation for retrieval-based question answering systems. In *Findings of the association for computational linguistics: Acl-ijcnlp 2021* (pp. 4276–4282). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.findings-acl.374` doi: 10.18653/v1/2021.findings-acl.374

Hsu, I., Huang, K.-H., Boschee, E., Miller, S., Natarajan, P., Chang, K.-W., ... others (2021). Degree: A data-efficient generative event extraction model. *arXiv preprint arXiv:2108.12724*.

Huang, L., Ji, H., & May, J. (2019, June). Cross-lingual multi-level adversarial transfer to enhance low-resource name tagging. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 3823–3833). Minneapolis, Minnesota: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N19-1383` doi: 10.18653/v1/N19-1383

Izacard, G., & Grave, E. (2021, April). Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: Main volume* (pp. 874–880). Online: Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.74

Izacard, G., Lewis, P., Lomeli, M., Hosseini, L., Petroni, F., Schick, T., … Grave, E. (2022). Atlas: Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*.

Jang, E., Gu, S., & Poole, B. (2017). Categorical reparameterization with gumbel-softmax. In *Proceedings of the 5th international conference on learning representations.*

Jawahar, G., Sagot, B., & Seddah, D. (2019, July). What does BERT learn about the structure of language? In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 3651–3657). Florence, Italy: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P19-1356` doi: 10.18653/v1/P19-1356

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., … Fung, P. (2023, mar). Survey of hallucination in natural language generation. *ACM Comput. Surv.*, *55*(12). doi: 10.1145/3571730

Jiang, Z., Araki, J., Ding, H., & Neubig, G. (2022, October). Understanding and improving zero-shot multi-hop reasoning in generative question answering. In *Proceedings of the 29th international conference on computational linguistics* (pp. 1765–1775). Gyeongju, Republic of Korea: International Committee on Computational Linguistics.

Joshi, P., Santy, S., Budhiraja, A., Bali, K., & Choudhury, M. (2020, July). The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 6282–6293). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-main.560` doi: 10.18653/v1/2020.acl-main.560

Joulin, A., Bojanowski, P., Mikolov, T., Jégou, H., & Grave, E. (2018, October-November). Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 2979–2984). Brussels, Belgium: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D18-1330` doi: 10.18653/v1/D18-1330

Judea, A., & Strube, M. (2016). Incremental global event extraction. In *Proceedings of the 26th international conference on computational linguistics.*

Kanayama, H., & Iwamoto, R. (2020, May). How universal are Universal Dependencies? exploiting syntax for multilingual clause-level sentiment detection. In *Proceedings of the twelfth language resources and evaluation conference* (pp. 4063–4073). Marseille, France: European Language Resources Association. Retrieved from `https://aclanthology.org/2020.lrec-1.500`

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... Amodei, D. (2020). Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., ... Yih, W.-t. (2020, November). Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 6769–6781). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.emnlp-main.550` doi: 10.18653/v1/2020.emnlp-main.550

Katiyar, A., & Cardie, C. (2017). Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In *Proceedings of the 55th annual meeting of the association for computational linguistics.*

Keung, P., Lu, Y., & Bhardwaj, V. (2019, November). Adversarial learning with contextual embeddings for zero-resource cross-lingual classification and NER. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 1355–1360). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D19-1138` doi: 10.18653/v1/D19-1138

Khandelwal, U., Levy, O., Jurafsky, D., Zettlemoyer, L., & Lewis, M. (2020). Generalization through memorization: Nearest neighbor language models. In *International conference on learning representations.* Retrieved from `https://openreview.net/forum?id=HklBjCEKvH`

Khashabi, D., Min, S., Khot, T., Sabharwal, A., Tafjord, O., Clark, P., & Hajishirzi, H. (2020, November). UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the association for computational linguistics: Emnlp 2020* (pp. 1896–1907). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.findings-emnlp.171` doi: 10.18653/v1/2020.findings-emnlp.171

Kim, Y. (2020, December). Deep active learning for sequence labeling based on diversity and uncertainty in gradient. In *Proceedings of the 2nd workshop on life-long learning for spoken language systems* (pp. 1–8). Suzhou, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.lifelongnlp-1.1`

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th international conference on learning representations.*

Kirkpatrick, S., Gelatt Jr, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, *220*(4598), 671–680.

Kittask, C., Milintsevich, K., & Sirts, K. (2020). Evaluating multilingual bert for estonian. *Volume*, *328*, 19–26.

Kondor, R., & Pan, H. (2016). The multiscale laplacian graph kernel. *Advances in neural information processing systems*, *29*.

Kondratyuk, D., & Straka, M. (2019, November). 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 2779–2795). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D19-1279` doi: 10.18653/v1/D19-1279

Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, *7*(1), 48–50.

Kudo, T. (2018, July). Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 66–75). Melbourne, Australia: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P18-1007` doi: 10.18653/v1/P18-1007

Kudo, T., & Richardson, J. (2018, November). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 conference on empirical methods in natural language processing: System demonstrations* (pp. 66–71). Brussels, Belgium: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D18-2012` doi: 10.18653/v1/D18-2012

Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Lai, V., Man, H., Ngo, L., Dernoncourt, F., & Nguyen, T. (2022, December). Multilingual SubEvent relation extraction: A novel dataset and structure induction method. In *Findings of the association for computational linguistics: Emnlp 2022* (pp. 5559–5570). Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.findings-emnlp.407`

Lai, V. D., Nguyen, M. V., Nguyen, T. H., & Dernoncourt, F. (2021). Graph learning regularization and transfer learning for few-shot event detection. In *Proceedings of the 44th international acm sigir conference on research and development in information retrieval* (pp. 2172–2176).

Lai, V. D., Nguyen, T. N., & Nguyen, T. H. (2020). Event detection: Gate diversity and syntactic importance scores for graph convolution neural networks. In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp).*

Lai, V. D., Veyseh, A. P. B., Nguyen, M. V., Dernoncourt, F., & Nguyen, T. H. (2022, October). MECI: A multilingual dataset for event causality identification. In *Proceedings of the 29th international conference on computational linguistics* (pp. 2346–2356). Gyeongju, Republic of Korea: International Committee on Computational Linguistics. Retrieved from `https://aclanthology.org/2022.coling-1.206`

Lange, L., Iurshina, A., Adel, H., & Strötgen, J. (2020a). Adversarial alignment of multilingual models for extracting temporal expressions from text. *arXiv preprint arXiv:2005.09392*.

Lange, L., Iurshina, A., Adel, H., & Strötgen, J. (2020b, July). Adversarial alignment of multilingual models for extracting temporal expressions from text. In *Proceedings of the 5th workshop on representation learning for nlp* (pp. 103–109). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.repl4nlp-1.14` doi: 10.18653/v1/2020.repl4nlp-1.14

Lauriola, I., & Moschitti, A. (2021). Answer sentence selection using local and global context in transformer models. In *Advances in information retrieval: 43rd european conference on ir research, ecir 2021, virtual event, march 28–april 1, 2021, proceedings, part i* (pp. 298–312).

Lewis, M., & Fan, A. (2019). Generative question answering: Learning to answer the whole question. In *International conference on learning representations.*

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., ... Zettlemoyer, L. (2020, July). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 7871–7880). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-main.703` doi: 10.18653/v1/2020.acl-main.703

Li, F., Peng, W., Chen, Y., Wang, Q., Pan, L., Lyu, Y., & Zhu, Y. (2020a). Event extraction as multi-turn question answering. In *Proceedings of the 2020 conference on empirical methods in natural language processing: Findings* (pp. 829–838).

Li, F., Peng, W., Chen, Y., Wang, Q., Pan, L., Lyu, Y., & Zhu, Y. (2020b, November). Event extraction as multi-turn question answering. In *Findings of the association for computational linguistics: Emnlp 2020* (pp. 829–838). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.findings-emnlp.73` doi: 10.18653/v1/2020.findings-emnlp.73

Li, Q., Anzaroot, S., Lin, W., Li, X., & Ji, H. (2011). Joint inference for cross-document information extraction. In *Proceedings of the 20th acm international conference on information and knowledge management.*

Li, Q., Ji, H., Hong, Y., & Li, S. (2014). Constructing information networks using one single model. In *Proceedings of the 2014 conference on empirical methods in natural language processing.*

Li, Q., Ji, H., & Huang, L. (2013a). Joint event extraction via structured prediction with global features. In *Proceedings of the 51th annual meeting of the association for computational linguistics.*

Li, Q., Ji, H., & Huang, L. (2013b, August). Joint event extraction via structured prediction with global features. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 73–82). Sofia, Bulgaria: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P13-1008`

Liao, S., & Grishman, R. (2011). Acquiring topic features to improve event extraction: in pre-selected and balanced collections. In *Ranlp.*

Lin, B., Yao, Z., Shi, J., Cao, S., Tang, B., Li, S., ... Hou, L. (2022, December). Dependency parsing via sequence generation. In *Findings of the association for computational linguistics: Emnlp 2022* (pp. 7339–7353). Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.findings-emnlp.543`

Lin, C.-Y. (2004, July). ROUGE: A package for automatic evaluation of summaries. In *Text summarization branches out* (pp. 74–81). Barcelona, Spain: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/W04-1013`

Lin, Y., Ji, H., Huang, F., & Wu, L. (2020a). A joint neural model for information extraction with global features. In *Proceedings of the 58th annual meeting of the association for computational linguistics.*

Lin, Y., Ji, H., Huang, F., & Wu, L. (2020b, July). A joint neural model for information extraction with global features. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 7999–8009). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-main.713` doi: 10.18653/v1/2020.acl-main.713

Liu, J., Chen, Y., Liu, K., Bi, W., & Liu, X. (2020, November). Event extraction as machine reading comprehension. In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 1641–1651). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.emnlp-main.128` doi: 10.18653/v1/2020.emnlp-main.128

Liu, J., Chen, Y., Liu, K., & Zhao, J. (2019a, November). Neural cross-lingual event detection with minimal parallel resources. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 738–748). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D19-1068` doi: 10.18653/v1/D19-1068

Liu, J., Chen, Y., Liu, K., & Zhao, J. (2019b, November). Neural cross-lingual event detection with minimal parallel resources. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 738–748). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://www.aclweb.org/anthology/D19-1068` doi: 10.18653/v1/D19-1068

Liu, M., Tu, Z., Zhang, T., Su, T., Xu, X., & Wang, Z. (2022). Ltp: A new active learning strategy for crf-based named entity recognition. *Neural Processing Letters*, 1–22.

Liu, X., He, P., Chen, W., & Gao, J. (2019, July). Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 4487–4496). Florence, Italy: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P19-1441` doi: 10.18653/v1/P19-1441

Liu, X., Huang, H., Shi, G., & Wang, B. (2022, May). Dynamic prefix-tuning for generative template-based event extraction. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 5216–5228). Dublin, Ireland: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.acl-long.358` doi: 10.18653/v1/2022.acl-long.358

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lowell, D., Lipton, Z. C., & Wallace, B. C. (2019, November). Practical obstacles to deploying active learning. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 21–30). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D19-1003` doi: 10.18653/v1/D19-1003

Lu, Y., Lin, H., Xu, J., Han, X., Tang, J., Li, A., … Chen, S. (2021a). Text2event: Controllable sequence-to-structure generation for end-to-end event extraction. *arXiv preprint arXiv:2106.09232*.

Lu, Y., Lin, H., Xu, J., Han, X., Tang, J., Li, A., … Chen, S. (2021b, August). Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)* (pp. 2795–2806). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.acl-long.217` doi: 10.18653/v1/2021.acl-long.217

Luan, Y., Wadden, D., He, L., Shah, A., Ostendorf, M., & Hajishirzi, H. (2019a). A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: Human language technologies.*

Luan, Y., Wadden, D., He, L., Shah, A., Ostendorf, M., & Hajishirzi, H. (2019b, June). A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 3036–3046). Minneapolis, Minnesota: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N19-1308` doi: 10.18653/v1/N19-1308

Ma, X., & Hovy, E. (2016). End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th annual meeting of the association for computational linguistics.*

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., & McClosky, D. (2014, June). The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: System demonstrations* (pp. 55–60). Baltimore, Maryland: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P14-5010` doi: 10.3115/v1/P14-5010

Margatina, K., Vernikos, G., Barrault, L., & Aletras, N. (2021). Active learning by acquiring contrastive examples. *arXiv preprint arXiv:2109.03764*.

Maynez, J., Narayan, S., Bohnet, B., & McDonald, R. (2020, July). On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 1906–1919). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-main.173` doi: 10.18653/v1/2020.acl-main.173

M'hamdi, M., Freedman, M., & May, J. (2019, November). Contextualized cross-lingual event trigger extraction with minimal resources. In *Proceedings of the 23rd conference on computational natural language learning (conll)* (pp. 656–665). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/K19-1061` doi: 10.18653/v1/K19-1061

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of the conference on neural information processing systems.*

Minh Tran, H., Phung, D., & Nguyen, T. H. (2021, August). Exploiting document structures and cluster consistencies for event coreference resolution. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)* (pp. 4840–4850). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.acl-long.374` doi: 10.18653/v1/2021.acl-long.374

Miwa, M., & Bansal, M. (2016). End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th annual meeting of the association for computational linguistics.*

Miwa, M., & Sasaki, Y. (2014). Modeling joint entity and relation extraction with table representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing.*

Mohit, B., Schneider, N., Bhowmick, R., Oflazer, K., & Smith, N. A. (2012, April). Recall-oriented learning of named entities in Arabic Wikipedia. In *Proceedings of the 13th conference of the European chapter of the association for computational linguistics* (pp. 162–173). Avignon, France: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/E12-1017`

Mohseni, M., & Tebbifakhr, A. (2019, 11–12 September). MorphoBERT: a Persian NER system with BERT and morphological analysis. In *Proceedings of the first international workshop on nlp solutions for under resourced languages (nsurl 2019) co-located with icnlsp 2019 - short papers* (pp. 23–30). Trento, Italy: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2019.nsurl-1.4`

Monge, G. (1781). Mémoire sur la théorie des déblais et des remblais. *Mem. Math. Phys. Acad. Royale Sci.*, 666–704.

Muller, B., Soldaini, L., Koncel-Kedziorski, R., Lind, E., & Moschitti, A. (2022, November). Cross-lingual open-domain question answering with answer sentence generation. In *Proceedings of the 2nd conference of the asia-pacific chapter of the association for computational linguistics and the 12th international joint conference on natural language processing (volume 1: Long papers)* (pp. 337–353). Online only: Association for Computational Linguistics.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., ... others (2021). Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Névéol, A., Robert, A., Anderson, R., Cohen, K. B., Grouin, C., Lavergne, T., ... Zweigenbaum, P. (2017). Clef ehealth 2017 multilingual information extraction task overview: Icd10 coding of death certificates in english and french. In *Clef (working notes)* (pp. 1–17).

Nghiem, M.-Q., Baylis, P., & Ananiadou, S. (2021, April). Paladin: an annotation tool based on active and proactive learning. In *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: System demonstrations* (pp. 238–243). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.eacl-demos.28` doi: 10.18653/v1/2021.eacl-demos.28

Ngo Trung, N., Phung, D., & Nguyen, T. H. (2021, August). Unsupervised domain adaptation for event detection using domain-specific adapters. In *Findings of the association for computational linguistics: Acl-ijcnlp 2021* (pp. 4015–4025). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.findings-acl.351` doi: 10.18653/v1/2021.findings-acl.351

Nguyen, D. Q., & Tuan Nguyen, A. (2020, November). PhoBERT: Pre-trained language models for Vietnamese. In *Findings of the association for computational linguistics: Emnlp 2020* (pp. 1037–1042). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.findings-emnlp.92` doi: 10.18653/v1/2020.findings-emnlp.92

Nguyen, M., C, K. K., Nguyen, T., Chadha, A., & Vu, T. (2023). Efficient fine-tuning large language models for knowledge-aware response planning. In *Ecml pkdd 2023.* Retrieved from `https://www.amazon.science/publications/efficient-fine-tuning-large-language-models-for-knowledge-aware-response-planning`

Nguyen, M. V., Lai, V. D., & Nguyen, T. H. (2021, June). Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 27–38). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.naacl-main.3` doi: 10.18653/v1/2021.naacl-main.3

Nguyen, M. V., Lai, V. D., Pouran Ben Veyseh, A., & Nguyen, T. H. (2021, April). Trankit: A light-weight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: System demonstrations* (pp. 80–90). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.eacl-demos.10` doi: 10.18653/v1/2021.eacl-demos.10

Nguyen, M. V., Lai, V. D., Veyseh, A. P. B., & Nguyen, T. H. (2021). Trankit: A light-weight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: System demonstrations.*

Nguyen, M. V., Min, B., Dernoncourt, F., & Nguyen, T. (2022a, July). Joint extraction of entities, relations, and events via modeling inter-instance and inter-label dependencies. In *Proceedings of the 2022 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 4363–4374). Seattle, United States: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.naacl-main.324` doi: 10.18653/v1/2022.naacl-main.324

Nguyen, M. V., Min, B., Dernoncourt, F., & Nguyen, T. (2022b, December). Learning cross-task dependencies for joint extraction of entities, events, event arguments, and relations. In *Proceedings of the 2022 conference on empirical methods in natural language processing* (pp. 9349–9360). Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.emnlp-main.634`

Nguyen, M. V., Ngo, N., Min, B., & Nguyen, T. (2022, July). FAMIE: A fast active learning framework for multilingual information extraction. In *Proceedings of the 2022 conference of the north american chapter of the association for computational linguistics: Human language technologies: System demonstrations* (pp. 131–139). Hybrid: Seattle, Washington + Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.naacl-demo.14` doi: 10.18653/v1/2022.naacl-demo.14

Nguyen, M. V., & Nguyen, T. H. (2021a). Improving cross-lingual transfer for event argument extraction with language-universal sentence structures. In *Proceedings of the sixth arabic natural language processing workshop (wanlp) at eacl 2021.*

Nguyen, M. V., & Nguyen, T. H. (2021b, April). Improving cross-lingual transfer for event argument extraction with language-universal sentence structures. In *Proceedings of the sixth arabic natural language processing workshop* (pp. 237–243). Kyiv, Ukraine (Virtual): Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.wanlp-1.27`

Nguyen, M. V., Nguyen, T. N., Min, B., & Nguyen, T. H. (2021, November). Crosslingual transfer learning for relation and event extraction via word category and class alignments. In *Proceedings of the 2021 conference on empirical methods in natural language processing* (pp. 5414–5426). Online and Punta Cana, Dominican Republic: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.emnlp-main.440` doi: 10.18653/v1/2021.emnlp-main.440

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., & Deng, L. (2016, November). Ms marco: A human generated machine reading comprehension dataset.

Nguyen, T. H., Cho, K., & Grishman, R. (2016a). Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics.*

Nguyen, T. H., Cho, K., & Grishman, R. (2016b, June). Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 conference of the north American chapter of the association for computational linguistics: Human language technologies* (pp. 300–309). San Diego, California: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N16-1034` doi: 10.18653/v1/N16-1034

Nguyen, T. H., Cho, K., & Grishman, R. (2016a). Joint event extraction via recurrent neural networks. In *Proceedings of the conference of the north american chapter of the association for computational linguistics: Human language technologies (naacl-hlt).*

Nguyen, T. H., & Grishman, R. (2015a). Event detection and domain adaptation with convolutional neural networks. In *The 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing.*

Nguyen, T. H., & Grishman, R. (2015b, July). Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 2: Short papers)* (pp. 365–371). Beijing, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P15-2060` doi: 10.3115/v1/P15-2060

Nguyen, T. H., & Grishman, R. (2015c). Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st naacl workshop on vector space modeling for nlp (vsm).*

Nguyen, T. H., & Grishman, R. (2018a). Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the aaai conference on artificial intelligence.*

Nguyen, T. H., & Grishman, R. (2018b). Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the association for the advancement of artificial intelligence (aaai).*

Nguyen, T. H., Sil, A., Dinu, G., & Florian, R. (2016). Toward mention detection robustness with recurrent neural networks. In *Proceedings of ijcai workshop on deep learning for artificial intelligence (dlai).*

Nguyen, T. M., & Nguyen, T. H. (2019a). One for all: Neural joint modeling of entities and events. In *Proceedings of the association for the advancement of artificial intelligence (aaai).*

Nguyen, T. M., & Nguyen, T. H. (2019b). One for all: Neural joint modeling of entities and events. In *Proceedings of the aaai conference on artificial intelligence.*

Ni, J., & Florian, R. (2019, November). Neural cross-lingual relation extraction based on bilingual word embedding mapping. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 399–409). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D19-1038` doi: 10.18653/v1/D19-1038

Nivre, J., de Marneffe, M.-C., Ginter, F., Hajič, J., Manning, C. D., Pyysalo, S., ... Zeman, D. (2020, May). Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the twelfth language resources and evaluation conference* (pp. 4034–4043). Marseille, France: European Language Resources Association. Retrieved from `https://aclanthology.org/2020.lrec-1.497`

Nothman, J., Ringland, N., Radford, W., Murphy, T., & Curran, J. R. (2012). Learning multilingual named entity recognition from Wikipedia. *Artificial Intelligence*, *194*, 151–175. Retrieved from `http://dx.doi.org/10.1016/j.artint.2012.03.006` doi: 10.1016/j.artint.2012.03.006

Pacheco Coelho, M. T., Pereira, E. B., Haynie, H. J., Rangel, T. F., Kavanagh, P., Kirby, K. R., ... others (2019). Drivers of geographical patterns of north american language diversity. *Proceedings of the Royal Society B*, *286*(1899), 20190242.

Paolini, G., Athiwaratkun, B., Krone, J., Ma, J., Achille, A., Anubhai, R., ... Soatto, S. (2021). Structured prediction as translation between augmented natural languages. In *9th international conference on learning representations, ICLR 2021.*

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002, July). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the association for computational linguistics* (pp. 311–318). Philadelphia, Pennsylvania, USA: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P02-1040` doi: 10.3115/1073083.1073135

Patwardhan, S., & Riloff, E. (2009). A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the annual meeting of the association for computational linguistics (acl).*

Peters, M. E., Ruder, S., & Smith, N. A. (2019a). To tune or not to tune? adapting pretrained representations to diverse tasks. In *Repl4nlp@acl.*

Peters, M. E., Ruder, S., & Smith, N. A. (2019b, August). To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th workshop on representation learning for nlp (repl4nlp-2019)* (pp. 7–14). Florence, Italy: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/W19-4302` doi: 10.18653/v1/W19-4302

Petroni, F., Rocktäschel, T., Riedel, S., Lewis, P., Bakhtin, A., Wu, Y., & Miller, A. (2019, November). Language models as knowledge bases? In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 2463–2473). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D19-1250` doi: 10.18653/v1/D19-1250

Pfeiffer, J., Rücklé, A., Poth, C., Kamath, A., Vulić, I., Ruder, S., ... Gurevych, I. (2020, October). AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations* (pp. 46–54). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.emnlp-demos.7` doi: 10.18653/v1/2020.emnlp-demos.7

Pfeiffer, J., Vulić, I., Gurevych, I., & Ruder, S. (2020, November). MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 7654–7673). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.emnlp-main.617` doi: 10.18653/v1/2020.emnlp-main.617

Poibeau, T., Saggion, H., Piskorski, J., & Yangarber, R. (2013). *Multi-source, multilingual information extraction and summarization*. Springer.

Pouran Ben Veyseh, A., Dernoncourt, F., Dou, D., & Nguyen, T. H. (2020, July). Exploiting the syntax-model consistency for neural relation extraction. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 8021–8032). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-main.715` doi: 10.18653/v1/2020.acl-main.715

Pouran Ben Veyseh, A., Ebrahimi, J., Dernoncourt, F., & Nguyen, T. (2022, December). MEE: A novel multilingual event extraction dataset. In *Proceedings of the 2022 conference on empirical methods in natural language processing* (pp. 9603–9613). Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.emnlp-main.652`

Pouran Ben Veyseh, A., Lai, V., Dernoncourt, F., & Nguyen, T. H. (2021, August). Unleash GPT-2 power for event detection. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)* (pp. 6271–6282). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.acl-long.490` doi: 10.18653/v1/2021.acl-long.490

Pouran Ben Veyseh, A., Nguyen, M. V., Dernoncourt, F., & Nguyen, T. (2022, July). MINION: a large-scale and diverse dataset for multilingual event detection. In *Proceedings of the 2022 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 2286–2299). Seattle, United States: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.naacl-main.166` doi: 10.18653/v1/2022.naacl-main.166

Pouran Ben Veyseh, A., Nguyen, M. V., Ngo Trung, N., Min, B., & Nguyen, T. H. (2021, November). Modeling document-level context for event detection via important context selection. In *Proceedings of the 2021 conference on empirical methods in natural language processing* (pp. 5403–5413). Online and Punta Cana, Dominican Republic: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.emnlp-main.439` doi: 10.18653/v1/2021.emnlp-main.439

Prim, R. C. (1957). Shortest connection networks and some generalizations. *The Bell System Technical Journal*, *36*(6), 1389–1401.

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., & Manning, C. D. (2020a). Stanza: A python natural language processing toolkit for many human languages. In *Acl.*

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., & Manning, C. D. (2020b, July). Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th annual meeting of the association for computational linguistics: System demonstrations* (pp. 101–108). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-demos.14` doi: 10.18653/v1/2020.acl-demos.14

Qu, M., Bengio, Y., & Tang, J. (2019). Gmnn: Graph markov neural networks. In *International conference on machine learning* (pp. 5241–5250).

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training.*

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, *1*(8), 9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... Liu, P. J. (2020a). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, *21*(140), 1-67. Retrieved from `http://jmlr.org/papers/v21/20-074.html`

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... Liu, P. J. (2020b). Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, *21*(1), 5485–5551.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... Liu, P. J. (2020c, jan). Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, *21*(1).

Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016, November). SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 conference on empirical methods in natural language processing* (pp. 2383–2392). Austin, Texas: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D16-1264` doi: 10.18653/v1/D16-1264

Raunak, V., Menezes, A., & Junczys-Dowmunt, M. (2021, June). The curious case of hallucinations in neural machine translation. In *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 1172–1183). Online: Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.92

Rebuffel, C., Roberti, M., Soulier, L., Scoutheeten, G., Cancelliere, R., & Gallinari, P. (2021). Controlling hallucinations at word level in data-to-text generation. *CoRR*, *abs/2102.02810*.

Richardson, M., Burges, C. J., & Renshaw, E. (2013, October). MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 193–203). Seattle, Washington, USA: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D13-1020`

Riedel, S., Chun, H.-W., Takagi, T., & Tsujii, J. (2009). A markov logic approach to bio-molecular event extraction. In *Proceedings of the bionlp 2009 workshop companion volume for shared task.*

Ro, Y., Lee, Y., & Kang, P. (2020, November). Multiˆ2OIE: Multilingual open information extraction based on multi-head attention with BERT. In *Findings of the association for computational linguistics: Emnlp 2020* (pp. 1107–1117). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.findings-emnlp.99` doi: 10.18653/v1/2020.findings-emnlp.99

Roberts, A., Raffel, C., & Shazeer, N. (2020a, November). How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 5418–5426). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.emnlp-main.437` doi: 10.18653/v1/2020.emnlp-main.437

Roberts, A., Raffel, C., & Shazeer, N. (2020b, November). How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 5418–5426). Online: Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.437

Roller, S., Dinan, E., Goyal, N., Ju, D., Williamson, M., Liu, Y., . . . Weston, J. (2021, April). Recipes for building an open-domain chatbot. In *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: Main volume* (pp. 300–325). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.eacl-main.24` doi: 10.18653/v1/2021.eacl-main.24

Roth, D., & Small, K. (2006). Margin-based active learning for structured output spaces. In *European conference on machine learning* (pp. 413–424).

Roth, D., & Yih, W.-t. (2004a). A linear programming formulation for global inference in natural language tasks. In *Proceedings of the eighth conference on computational natural language learning.*

Roth, D., & Yih, W.-t. (2004b, May 6 - May 7). A linear programming formulation for global inference in natural language tasks. In *Proceedings of the eighth conference on computational natural language learning (CoNLL-2004) at HLT-NAACL 2004* (pp. 1–8). Boston, Massachusetts, USA: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/W04-2401`

Sahu, S. K., Christopoulou, F., Miwa, M., & Ananiadou, S. (2019). Inter-sentence relation extraction with document-level graph convolutional neural network. In *Proceedings of the annual meeting of the association for computational linguistics (acl).*

Santos, C. d., & Guimaraes, V. (2015). Boosting named entity recognition with neural character embeddings. In *Proceedings of the fifth named entity workshop.*

Saxena, A., Chakrabarti, S., & Talukdar, P. (2021, August). Question answering over temporal knowledge graphs. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)* (pp. 6663–6676). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.acl-long.520` doi: 10.18653/v1/2021.acl-long.520

Scutari, M., Graafland, C. E., & Gutiérrez, J. M. (2019). Who learns better bayesian network structures: Accuracy and speed of structure learning algorithms. *International Journal of Approximate Reasoning*, *115*, 235–253.

Sener, O., & Savarese, S. (2017). Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Settles, B. (2009). Active learning literature survey.

Settles, B., & Craven, M. (2008, October). An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the 2008 conference on empirical methods in natural language processing* (pp. 1070–1079). Honolulu, Hawaii: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D08-1112`

Severyn, A., & Moschitti, A. (2015). Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international acm sigir conference on research and development in information retrieval* (pp. 373–382).

Shelmanov, A., Puzyrev, D., Kupriyanova, L., Belyakov, D., Larionov, D., Khromov, N., ... Panchenko, A. (2021, April). Active learning for sequence tagging with deep pre-trained models and Bayesian uncertainty estimates. In *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: Main volume* (pp. 1698–1712). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.eacl-main.145` doi: 10.18653/v1/2021.eacl-main.145

Shen, Y., Yun, H., Lipton, Z., Kronrod, Y., & Anandkumar, A. (2017b, August). Deep active learning for named entity recognition. In *Proceedings of the 2nd workshop on representation learning for NLP* (pp. 252–256). Vancouver, Canada: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/W17-2630` doi: 10.18653/v1/W17-2630

Shen, Y., Yun, H., Lipton, Z. C., Kronrod, Y., & Anandkumar, A. (2017a). Deep active learning for named entity recognition. *arXiv preprint arXiv:1707.05928*.

Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K., & Borgwardt, K. (2009). Efficient graphlet kernels for large graph comparison. In *Artificial intelligence and statistics* (pp. 488–495).

Shishtla, P. M., Gali, K., Pingali, P., & Varma, V. (2008). Experiments in telugu ner: A conditional random field approach. In *Proceedings of the ijcnlp-08 workshop on named entity recognition for south and south east asian languages.*

Shuster, K., Poff, S., Chen, M., Kiela, D., & Weston, J. (2021a, November). Retrieval augmentation reduces hallucination in conversation. In *Findings of the association for computational linguistics: Emnlp 2021* (pp. 3784–3803). Punta Cana, Dominican Republic: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.findings-emnlp.320` doi: 10.18653/v1/2021.findings-emnlp.320

Shuster, K., Poff, S., Chen, M., Kiela, D., & Weston, J. (2021b, November). Retrieval augmentation reduces hallucination in conversation. In *Findings of the association for computational linguistics: Emnlp 2021* (pp. 3784–3803). Punta Cana, Dominican Republic: Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.320

Sinkhorn, R., & Knopp, P. (1967). Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, *21*(2), 343–348.

Sobhana, N., Mitra, P., & Ghosh, S. (2010). Conditional random field based named entity recognition in geological text. *International Journal of Computer Applications*, *1*(3), 143–147.

Søgaard, A. (2022). Should we ban english nlp for a year? In *Proceedings of the 2022 conference on empirical methods in natural language processing* (pp. 5254–5260).

Soltan, S., Ananthakrishnan, S., FitzGerald, J. G. M., Gupta, R., Hamza, W., Khan, H., . . . Natarajan, P. (2022). Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model. *arXiv*.

Song, Z., Bies, A., Strassel, S., Riese, T., Mott, J., Ellis, J., . . . Ma, X. (2015, June). From light to rich ERE: Annotation of entities, relations, and events. In *Proceedings of the the 3rd workshop on EVENTS: Definition, detection, coreference, and representation* (pp. 89–98). Denver, Colorado: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/W15-0812` doi: 10.3115/v1/W15-0812

Straka, M. (2018a). Udpipe 2.0 prototype at conll 2018 ud shared task. In *Conll.*

Straka, M. (2018b, October). UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies* (pp. 197–207). Brussels, Belgium: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/K18-2020` doi: 10.18653/v1/K18-2020

Straka, M., Hajič, J., & Straková, J. (2016, May). UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the tenth international conference on language resources and evaluation (LREC'16)* (pp. 4290–4297). Portorož, Slovenia: European Language Resources Association (ELRA). Retrieved from `https://aclanthology.org/L16-1680`

Straka, M., & Straková, J. (2017, August). Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 shared task: Multilingual parsing from raw text to universal dependencies* (pp. 88–99). Vancouver, Canada: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/K17-3009` doi: 10.18653/v1/K17-3009

Subburathinam, A., Lu, D., Ji, H., May, J., Chang, S.-F., Sil, A., & Voss, C. (2019, November). Cross-lingual structure transfer for relation and event extraction. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 313–325). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D19-1030` doi: 10.18653/v1/D19-1030

Sun, C., Gong, Y., Wu, Y., Gong, M., Jiang, D., Lan, M., ... Duan, N. (2019). Joint type inference on entities and relations via graph convolutional networks. In *Proceedings of the 57th annual meeting of the association for computational linguistics.*

Sun, X., Lin, X., Shen, S., & Hu, Z. (2017). High-resolution remote sensing data classification over urban areas using random forest ensemble and fully connected conditional random field. *ISPRS International Journal of Geo-Information*, *6*(8), 245.

Taghizadeh, N., & Faili, H. (2020). Cross-lingual adaptation using universal dependencies. *arXiv preprint arXiv:2003.10816*.

Tang, H., Chen, K., & Jia, K. (2020). Unsupervised domain adaptation via structurally regularized deep clustering. In *Proceedings of the conference on computer vision and pattern recognition (cvpr).*

Tenney, I., Das, D., & Pavlick, E. (2019, July). BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 4593–4601). Florence, Italy: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P19-1452` doi: 10.18653/v1/P19-1452

Tian, Y., Song, Y., Ao, X., Xia, F., Quan, X., Zhang, T., & Wang, Y. (2020, July). Joint Chinese word segmentation and part-of-speech tagging via two-way attentions of auto-analyzed knowledge. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 8286–8296). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.acl-main.735` doi: 10.18653/v1/2020.acl-main.735

Tjong Kim Sang, E. F. (2002). Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th conference on natural language learning 2002 (CoNLL-2002).* Retrieved from `https://aclanthology.org/W02-2024`

Tjong Kim Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on natural language learning at HLT-NAACL 2003* (pp. 142–147). Retrieved from `https://aclanthology.org/W03-0419`

Tsai, H., Riesa, J., Johnson, M., Arivazhagan, N., Li, X., & Archer, A. (2019, November). Small and practical BERT models for sequence labeling. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 3632–3636). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D19-1374` doi: 10.18653/v1/D19-1374

Üstün, A., Bisazza, A., Bouma, G., & van Noord, G. (2020, November). UDapter: Language adaptation for truly Universal Dependency parsing. In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 2302–2315). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.emnlp-main.180` doi: 10.18653/v1/2020.emnlp-main.180

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, *9*(11).

Van Laarhoven, P. J., & Aarts, E. H. (1987). Simulated annealing. In *Simulated annealing: Theory and applications* (pp. 7–15). Springer.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … Polosukhin, I. (2017). Attention is all you need. In I. Guyon et al. (Eds.), *Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc. Retrieved from `https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`

Venugopal, D., Chen, C., Gogate, V., & Ng, V. (2014). Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features. In *Proceedings of the 2014 conference on empirical methods in natural language processing.*

Veyseh, A. P. B., Dernoncourt, F., Dou, D., & Nguyen, T. H. (2020a). Exploiting the syntax-model consistency for neural relation extraction. In *Proceedings of the annual meeting of the association for computational linguistics (acl).*

Veyseh, A. P. B., Dernoncourt, F., Dou, D., & Nguyen, T. H. (2020b). Exploiting the syntax-model consistency for neural relation extraction. In *Proceedings of the annual meeting of the association for computational linguistics (acl).*

Veyseh, A. P. B., Dernoncourt, F., Thai, M., Dou, D., & Nguyen, T. H. (2020a). Multi-view consistency for relation extraction via mutual information and structure prediction. In *Proceedings of the association for the advancement of artificial intelligence (aaai).*

Veyseh, A. P. B., Dernoncourt, F., Thai, M., Dou, D., & Nguyen, T. H. (2020b). Multi-view consistency for relation extraction via mutual information and structure prediction. In *Proceedings of the association for the advancement of artificial intelligence (aaai).*

Veyseh, A. P. B., Nguyen, M. V., Min, B., & Nguyen, T. H. (2021). Augmenting open-domain event detection with synthetic data from gpt-2. In *Joint european conference on machine learning and knowledge discovery in databases* (pp. 644–660).

Veyseh, A. P. B., Nguyen, T. N., & Nguyen, T. H. (2020a). Graph transformer networks with syntactic and semantic structures for event argument extraction. In *Proceedings of the findings of the 2020 conference on empirical methods in natural language processing (emnlp findings).*

Veyseh, A. P. B., Nguyen, T. N., & Nguyen, T. H. (2020b). Graph transformer networks with syntactic and semantic structures for event argument extraction. In *Proceedings of the conference on empirical methods in natural language processing: Findings (emnlp).*

Vishwanathan, S., Borgwardt, K. M., Schraudolph, N. N., et al. (2006). Fast computation of graph kernels. In *Nips* (Vol. 19, pp. 131–138).

Wadden, D., Wennberg, U., Luan, Y., & Hajishirzi, H. (2019a). Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 57th annual meeting of the association for computational linguistics.*

Wadden, D., Wennberg, U., Luan, Y., & Hajishirzi, H. (2019b, November). Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 5784–5789). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D19-1585` doi: 10.18653/v1/D19-1585

Walker, C., Strassel, S., Medero, J., & Maeda, K. (2006). Ace 2005 multilingual training corpus. In *Technical report, linguistic data consortium.*

Wang, C., & Sennrich, R. (2020, July). On exposure bias, hallucination and domain shift in neural machine translation. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 3544–3552). Online: Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.326

Wang, D., & Shang, Y. (2014). A new active labeling method for deep learning. In *2014 international joint conference on neural networks (ijcnn)* (pp. 112–119).

Wang, S., Yu, M., Chang, S., Sun, L., & Huang, L. (2022, May). Query and extract: Refining event extraction as type-oriented binary decoding. In *Findings of the association for computational linguistics: Acl 2022* (pp. 169–182). Dublin, Ireland: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.findings-acl.16` doi: 10.18653/v1/2022.findings-acl.16

Wang, W., Bao, H., Huang, S., Dong, L., & Wei, F. (2021, August). MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the association for computational linguistics: Acl-ijcnlp 2021* (pp. 2140–2151). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.findings-acl.188` doi: 10.18653/v1/2021.findings-acl.188

Wang, W., Yang, N., Wei, F., Chang, B., & Zhou, M. (2017). Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 189–198).

Wang, X., Wang, Z., Han, X., Liu, Z., Li, J., Li, P., ... Ren, X. (2019). Hmeae: Hierarchical modular event argument extraction. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp).*

Wang, Y., Mishra, S., Alipoormolabashi, P., Kordi, Y., Mirzaei, A., Naik, A., ... Shen, X. (2022, December). Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 conference on empirical methods in natural language processing* (pp. 5085–5109). Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2022.emnlp-main.340`

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., ... Fedus, W. (2022). Emergent abilities of large language models. *Transactions on Machine Learning Research.* (Survey Certification)

Weischedel, R., Palmer, M., Marcus, M., Hovy, E., Pradhan, S., Ramshaw, L., ... others (2013). Ontonotes release 5.0. *Linguistic Data Consortium.*

Weston, J., Bordes, A., Chopra, S., Rush, A. M., Van Merriënboer, B., Joulin, A., & Mikolov, T. (2015). Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698.*

Wiechmann, M., Yimam, S. M., & Biemann, C. (2021, June). ActiveAnno: General-purpose document-level annotation tool with active learning integration. In *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: Human language technologies: Demonstrations* (pp. 99–105). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.naacl-demos.12` doi: 10.18653/v1/2021.naacl-demos.12

Wiseman, S., & Rush, A. M. (2016, November). Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 conference on empirical methods in natural language processing* (pp. 1296–1306). Austin, Texas: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D16-1137` doi: 10.18653/v1/D16-1137

Xiao, Y., & Wang, W. Y. (2021, April). On hallucination and predictive uncertainty in conditional language generation. In *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: Main volume* (pp. 2734–2744). Online: Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.236

Xie, S., Zheng, Z., Chen, L., & Chen, C. (2018). Learning semantic representations for unsupervised domain adaptation. In *Proceedings of the international conference on machine learning (icml).*

Xu, J., Wang, Y., Tang, D., Duan, N., Yang, P., Zeng, Q., ... Sun, X. (2019). Asking clarification questions in knowledge-based question answering. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 1618–1629).

Xu, K., Zhou, Z., Hao, T., & Liu, W. (2017). A bidirectional lstm and conditional random fields approach to medical named entity recognition. In *International conference on advanced intelligent systems and informatics* (pp. 355–365).

Yan, H., Gui, T., Dai, J., Guo, Q., Zhang, Z., & Qiu, X. (2021, August). A unified generative framework for various NER subtasks. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)* (pp. 5808–5822). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.acl-long.451` doi: 10.18653/v1/2021.acl-long.451

Yang, B., & Mitchell, T. M. (2016a). Joint extraction of events and entities within a document context. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: Human language technologies.*

Yang, B., & Mitchell, T. M. (2016b, June). Joint extraction of events and entities within a document context. In *Proceedings of the 2016 conference of the north American chapter of the association for computational linguistics: Human language technologies* (pp. 289–299). San Diego, California: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N16-1033` doi: 10.18653/v1/N16-1033

Yang, H. (2019). Bert meets chinese word segmentation. *arXiv preprint arXiv:1909.09292*.

Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K., ... Lin, J. (2019, June). End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics (demonstrations)* (pp. 72–77). Minneapolis, Minnesota: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N19-4013` doi: 10.18653/v1/N19-4013

Yang, Y., Yih, W.-t., & Meek, C. (2015a). Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 2013–2018).

Yang, Y., Yih, W.-t., & Meek, C. (2015b, September). WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 2013–2018). Lisbon, Portugal: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D15-1237` doi: 10.18653/v1/D15-1237

Yoon, S., Dernoncourt, F., Kim, D. S., Bui, T., & Jung, K. (2019). A compare-aggregate model with latent clustering for answer selection. In *Proceedings of the 28th acm international conference on information and knowledge management* (pp. 2093–2096).

Yu, X., & Lam, W. (2010a). Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In *Proceedings of the 23th international conference on computational linguistics.*

Yu, X., & Lam, W. (2010b, August). Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In *Coling 2010: Posters* (pp. 1399–1407). Beijing, China: Coling 2010 Organizing Committee. Retrieved from `https://aclanthology.org/C10-2160`

Yuan, H., & Ji, S. (2020). Structpool: Structured graph pooling via conditional random fields. In *Proceedings of the 8th international conference on learning representations.*

Yuan, M., Lin, H.-T., & Boyd-Graber, J. (2020, November). Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 7935–7948). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2020.emnlp-main.637` doi: 10.18653/v1/2020.emnlp-main.637

Zaugg, I. A. (2020). Digital inequality and language diversity: An ethiopic case study. *Digital inequalities in the global south*, 247–267.

Zea, J. L. C., Luna, J. E. O., Thorne, C., & Glavaš, G. (2016). Spanish ner with word representations and conditional random fields. In *Proceedings of the sixth named entity workshop* (pp. 34–40).

Zeman, D., Nivre, J., Abrams, M., Ackermann, E., Aepli, N., Aghaei, H., ... Zhuravleva, A. (2020). *Universal dependencies 2.7.* (LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University)

Zeman, D., Nivre, J., Abrams, M., Aepli, N., Agić, Ž., Ahrenberg, L., ... Zhu, H. (2019). *Universal dependencies 2.5.* Retrieved from `http://hdl.handle.net/11234/1-3105` (LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University)

Zhang, J., Qin, Y., Zhang, Y., Liu, M., & Ji, D. (2019). Extracting entities and events as a single task using a transition-based neural model. In *Proceedings of the twenty-eighth international joint conference on artificial intelligence.*

Zhang, J., Zhao, Y., Saleh, M., & Liu, P. (2020, 13–18 Jul). PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In H. D. III & A. Singh (Eds.), *Proceedings of the 37th international conference on machine learning* (Vol. 119, pp. 11328–11339). PMLR.

Zhang*, T., Kishore*, V., Wu*, F., Weinberger, K. Q., & Artzi, Y. (2020). Bertscore: Evaluating text generation with bert. In *International conference on learning representations.*

Zhang, Z., & Ji, H. (2021a). Abstract meaning representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 39–49).

Zhang, Z., & Ji, H. (2021b, June). Abstract Meaning Representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 39–49). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.naacl-main.4` doi: 10.18653/v1/2021.naacl-main.4

Zhang, Z., Vu, T., Gandhi, S., Chadha, A., & Moschitti, A. (2022a). Wdrass: A web-scale dataset for document retrieval and answer sentence selection. In *Proceedings of the 31st acm international conference on information & knowledge management* (pp. 4707–4711).

Zhang, Z., Vu, T., Gandhi, S., Chadha, A., & Moschitti, A. (2022b). Wdrass: A web-scale dataset for document retrieval and answer sentence selection. In *Proceedings of the 31st acm international conference on information and knowledge management* (p. 4707–4711). Association for Computing Machinery.

Zhao, Z., Cohen, S. B., & Webber, B. (2020, November). Reducing quantity hallucinations in abstractive summarization. In *Findings of the association for computational linguistics: Emnlp 2020* (pp. 2237–2249). Online: Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.203

Zheng, S., Wang, F., Bao, H., Hao, Y., Zhou, P., & Xu, B. (2017a). Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th annual meeting of the association for computational linguistics.*

Zheng, S., Wang, F., Bao, H., Hao, Y., Zhou, P., & Xu, B. (2017b, July). Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 1227–1236). Vancouver, Canada: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P17-1113` doi: 10.18653/v1/P17-1113

Zhou, C., Neubig, G., Gu, J., Diab, M., Guzmán, F., Zettlemoyer, L., & Ghazvininejad, M. (2021a, August). Detecting hallucinated content in conditional neural sequence generation. In *Findings of the association for computational linguistics: Acl-ijcnlp 2021* (pp. 1393–1404). Online: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/2021.findings-acl.120` doi: 10.18653/v1/2021.findings-acl.120

Zhou, C., Neubig, G., Gu, J., Diab, M., Guzmán, F., Zettlemoyer, L., & Ghazvininejad, M. (2021b, August). Detecting hallucinated content in conditional neural sequence generation. In *Findings of the association for computational linguistics: Acl-ijcnlp 2021* (pp. 1393–1404). Online: Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.120

Zhou, G., Su, J., Zhang, J., & Zhang, M. (2005a). Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting of the association for computational linguistics.*

Zhou, G., Su, J., Zhang, J., & Zhang, M. (2005b, June). Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (ACL'05)* (pp. 427–434). Ann Arbor, Michigan: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P05-1053` doi: 10.3115/1219840.1219893

Zhu, X. (2020). Cross-lingual word sense disambiguation using mbert embeddings with syntactic dependencies. *arXiv preprint arXiv:2012.05300*.