WEAKLY SUPERVISED TEXT MINING WITH TEXT-RICH NETWORKS

BY

XINYANG ZHANG

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois Urbana-Champaign, 2024

Urbana, Illinois

Doctoral Committee:

Professor Jiawei Han, Chair and Director of Research
Professor Hari Sundaram
Professor Hanghang Tong
Dr. Xin Luna Dong, Meta

ABSTRACT

The advent of the information age has brought about an unprecedented surge in the volume of data available at our fingertips. A significant portion of this data is manifested in the form of *semi-structured* text corpora, which are characterized by the interplay between unstructured text and structured metadata. These corpora are ubiquitous across various domains, presenting unique challenges for knowledge discovery and data mining. For instance, social media platforms are a complex blend of users, user-generated content, and the intricate web of interactions between them. Similarly, academic publication databases comprise a rich tapestry of published content, author information, and venue details. The sheer scale and heterogeneity of these semi-structured text corpora necessitate the development of intelligent and efficient techniques to extract, organize, and summarize the valuable knowledge embedded within them. Traditional approaches to text mining often rely heavily on extensive human annotation or manually curated knowledge bases, which are not only time-consuming and expensive but also difficult to adapt to new domains. To fully harness the potential of these vast information resources, we propose a novel approach that leverages the inherent structure of the data itself.

We introduce a framework based on *text-rich networks*, which effectively integrates unstructured documents with structured metadata to create a comprehensive representation for data mining. By interconnecting documents based on shared attributes, such as linking research papers authored by the same individual, we construct a unified network that captures both the textual content and the relationships between entities. This approach opens up new possibilities for knowledge discovery, enabling us to uncover valuable patterns and insights that may be difficult to detect using traditional methods. The text-rich network provides a natural encoding of document relevance, allowing us to develop weakly supervised text mining applications that require minimal human intervention. This methodology reduces the reliance on large amounts of labeled data, making it more scalable and adaptable to various domains. Through the use of text-rich networks, we aim to enhance the efficiency and effectiveness of knowledge discovery in semi-structured text corpora, facilitating data-driven insights and innovation in the field of text mining.

My research consists of two main areas of investigation: (1) construction and consolidation of a text-rich network, and (2) mining of a text-rich network.

In the first area, we focus on building a robust text-rich network structure that relies on the availability of structured information alongside text. When such information is unavailable

or incomplete, we investigate weakly supervised methods to extract structured information.

1. Open-World Attribute Value Extraction: We propose a novel approach for open-world attribute value extraction, which enables us to identify entities and attributes that facilitate the construction of a comprehensive text-rich network. This method addresses the challenges of incomplete or missing structured information in semi-structured text corpora.

In the second area, we explore various methods built on top of a fully constructed text-rich network for fundamental text representation and weakly supervised text mining applications.

2. Language Model Pre-training with Text-Rich Networks: We introduce a novel language model pre-training approach that utilizes the text-rich network structure to capture more meaningful representations of text data. By incorporating the network's contextual information during the pre-training process, we aim to improve the effectiveness of downstream natural language processing tasks.

3. Text-Rich Network-Driven Weakly Supervised Text Classification: We develop a framework that leverages the rich structural information embedded in the text-rich network to enhance the performance of text classification tasks. This approach reduces the reliance on large amounts of labeled data, making it more adaptable to real-world scenarios.

Together, these components create a cohesive framework for weakly supervised text mining with text-rich networks. Our open-world attribute value extraction method strengthens the construction and consolidation of text-rich networks, while our text classification and language model pre-training approaches demonstrate the power of mining these networks for various applications. By integrating these techniques, we establish a comprehensive methodology that enables efficient and effective knowledge discovery in semi-structured text corpora, reducing the reliance on human annotation and making it more adaptable to real-world scenarios.

To my loved ones and younger self.

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my advisor, Professor Jiawei Han, for his unwavering support, guidance, and mentorship throughout my doctoral journey. Professor Han exemplifies the qualities of a true researcher, demonstrating an unparalleled dedication and passion for his work. His commitment to addressing "real problems, real data, and real solutions" has been a guiding principle that I strive to emulate in my own research endeavors. Beyond his academic brilliance, Professor Han's kindness and empathy have been a beacon of light during challenging times. When faced with the unprecedented difficulties brought about by the Covid-19 pandemic, Professor Han provided steadfast support and understanding as I navigated life, research, and mental health challenges. His compassion and encouragement were instrumental in helping me persevere through these trying circumstances. I am truly fortunate to have had the opportunity to learn from and work alongside such an exceptional mentor and human being.

I would also like to extend my sincere gratitude to the esteemed members of my Thesis Committee: Doctor Xin Luna Dong, Professor Hari Sundaram, and Professor Hanghang Tong. Despite their busy schedules, they generously offered their time and support throughout the thesis process. Their invaluable feedback and constructive critiques have greatly enhanced the quality of my thesis. I am truly appreciative of their willingness to serve on my committee and their commitment to supporting my academic journey. Their guidance and encouragement have been invaluable in bringing my thesis to fruition.

I would like to express my heartfelt gratitude to all of my collaborators who have contributed to the research presented in this thesis. Their expertise, insights, and hard work have been instrumental in advancing our projects and pushing the boundaries of our knowledge. I have been fortunate to work alongside such talented and dedicated individuals, and I am truly grateful for their collaboration and friendship. They are Jingbo Shang, Chenwei Zhang, Aravind Sankar, Ahmed El-Kishky, Yury Malkov, Jiayun Zhang, Liyuan Liu, Sha Li, Omar Florez, Serim Park, Brian McWilliams, Xian Li, Christos Faloutsos, Bowen Jin, Dheeraj Mekala.

I want to express my heartfelt gratitude to the incredible friends I have had the privilege of meeting throughout my journey. These friendships have enriched my life in countless ways, providing me with a support system that spans across the miles. I am forever grateful for the memories we've created, the bonds we've forged, and the unwavering support they've shown me. To my best friends, thank you for being an integral part of my journey and for making

my graduate experience an unforgettable one. They are Tianyu Ao, Shiwei Fu, Kayla Lin, Xiaolu Qi, Cheng Ding, Wei-Chen Lin, Anqing Zheng, Zichen Zhang, Jinhao Lei, Kuida Liu, Xingjian Zhen, Rui Wang, Yuying Chen, Fengjia Chen, Ziwei Wang, Jingwen Cai.

I am fortunate to be part of the Data Mining Group and would like to extend my warmest thanks to my fellow researchers and friends here. Our time together at the Siebel Center and in the vibrant community of Urbana-Champaign has been an unforgettable part of my graduate journey. From engaging in thought-provoking discussions and collaborating on challenging projects to enjoying spring outings and hotpots, the memories we've created together will forever be cherished. They are Jatin Arora, Suyu Ge, Sayar Ghosh Roy, Fang Guo, Xiaotao Gu, Jiaxin Huang, Yizhu Jiao, Priyanka Kargupta, Tanay Komarlu, Qi Li, Ruiliang Lyu, Yuning Mao, Yu Meng, Siru Ouyang, Chanyong Park, Wenda Qiu, Yanru Qu, Karthik Venkat Ramanan, Jiaming Shen, Yu Shi, Xuan Wang, Jinfeng Xiao, Yiqing Xie, Carl Yang, Xiao Yu, Chao Zhang, Yu Zhang, Yunyi Zhang, Ming Zhong, Qi Zhu, Shi Zhi, Honglei Zhuang, and Wanzheng Zhu.

Last but not least, my deepest appreciation goes to my parents, Lihong Yi and Yongxun Zhang. Their unconditional love and unwavering support have been the bedrock of my journey.

Research in this thesis was supported in part by US DARPA KAIROS Program No. FA8750-19-2-1004 and INCAS Program No. HR001121C0165, National Science Foundation IIS-19-56151, and the Molecule Maker Lab Institute: An AI Research Institutes program supported by NSF under Award No. 2019897, and the Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE) by NSF under Award No. 2118329. Any opinions, findings, and conclusions or recommendations expressed herein are those of the authors and do not necessarily represent the views, either expressed or implied, of DARPA or the U.S. Government.

CONTENTS

Chapter 1: INTRODUCTION

1.1 OVERVIEW AND MOTIVATION

In recent years, the rapid growth of digital technologies has led to an explosion of information, with a substantial portion of this data being generated in the form of *semi-structured text corpora*. These corpora, consisting of a combination of unstructured text and structured metadata, are prevalent across various domains, from social media and online forums to scientific publications and enterprise databases. The inherent complexity and diversity of these semi-structured text corpora present significant challenges for effective knowledge discovery and data mining.

One prominent example of semi-structured text corpora is social media platforms like Twitter and Facebook. These platforms generate vast amounts of user-generated content, including posts, comments, and user interactions, along with structured metadata like user profiles, timestamps, and geolocation information. Similarly, in the academic domain, publication databases store a wealth of information in the form of research papers, authors, affiliations, and citation networks. The ability to efficiently extract and analyze the valuable knowledge hidden within these corpora can drive innovation, inform decision-making, and uncover new insights across various fields.

However, traditional approaches to text mining often struggle to handle the unique characteristics of semi-structured text corpora effectively. Many of these methods rely heavily on manual annotation or pre-existing knowledge bases, which can be costly, time-consuming, and difficult to scale to large datasets. Furthermore, these approaches often treat unstructured text and structured metadata as separate entities, failing to fully exploit their rich connections and interactions. As a result, there is a pressing need for novel techniques that can seamlessly integrate both the textual content and the underlying network structure to enable more effective and efficient knowledge discovery in semi-structured text corpora.

To address these challenges, we propose a framework based on *text-rich networks* (TRNs), which provides a unified representation of semi-structured text corpora by connecting documents through their shared attributes and relationships. By leveraging the inherent structure of the data itself, our approach reduces the reliance on extensive human annotation and enables weakly supervised learning techniques that can adapt to various domains with minimal effort. Text-rich networks allow us to capture the complex interactions between textual content and metadata, providing a more comprehensive and nuanced understanding of the underlying knowledge within the corpora.

The power of text-rich networks lies in their ability to encode document relevance naturally, enabling the development of advanced text-mining applications that require minimal human intervention. For instance, by linking research papers based on their authors, citations, and publication venues, we can uncover hidden patterns and trends in scientific literature that may be difficult to detect using traditional methods. Similarly, by analyzing the structure of user interactions and content sharing on social media platforms, we can gain valuable insights into user behavior, community dynamics, and information propagation.

Through the use of text-rich networks, we aim to advance the field of text mining by providing a powerful and flexible framework for knowledge discovery in semi-structured text corpora. By harnessing the full potential of these information-rich resources, we can unlock new opportunities for data-driven decision-making, accelerate scientific discovery, and drive innovation across a wide range of domains.

1.2 TEXT-RICH NETWORKS

We now provide a formal definition of a text-rich network, which serves as the foundation for our weakly supervised mining approach in semi-structured text corpora.

Definition 1.1 (**Text-Rich Network**). *A text-rich network $\mathcal{N}$ is a tuple $(D, V, E, \phi, \psi, \omega)$, where:*

- *D is a set of textual documents.*
- *V is a set of nodes, which is partitioned into two disjoint subsets: V_t, the set of textual nodes, and V_a, the set of auxiliary nodes. Textual nodes represent documents with raw text content, while auxiliary nodes represent entities, metadata, or other relevant information without raw text content.*
- *$E \subseteq V \times V$ is a set of edges connecting nodes in V. Each edge represents a relationship or interaction between the connected nodes.*
- *$\phi : V \to \mathcal{T}_v$ is a node type mapping that assigns each node to a type in the node type set $\mathcal{T}_v$.*
- *$\psi : E \to \mathcal{T}_e$ is an edge type mapping that assigns each edge to a type in the edge type set $\mathcal{T}_e$.*
- *$\omega : V_t \to D$ is a bijective mapping that associates each textual node with its corresponding document in D. This mapping ensures that every document in D is represented by one textual node in V_t.*

To illustrate the concept of a text-rich network, let us consider an example constructed from an online book review website. In this context, the textual documents D would consist of book descriptions and user reviews. The node set V would include textual nodes V_t representing the book descriptions and reviews, as well as auxiliary nodes V_a representing authors, publishers, and other relevant entities. The edge set E would capture the relationships between these nodes, such as authorship edges connecting authors to their books, and publishing edges connecting books to their respective publishers. The node type mapping ϕ would assign types to the nodes, such as "book," "author," or "publisher," allowing for the differentiation between various entities in the network. Similarly, the edge type mapping ψ would assign types to the edges, such as "authored" for authorship relationships or "published" for publishing relationships. Finally, the mapping ω would associate each textual node (book description or review) with its corresponding document in D.

The text-rich network definition 1.1 provides a flexible and expressive framework for modeling semi-structured text corpora. By incorporating both textual content and structured metadata, text-rich networks enable the capture of complex relationships and interactions between documents, entities, and other relevant information. This unified representation lays the groundwork for developing weakly supervised mining techniques that can effectively harness the data's rich structure to uncover valuable insights and knowledge.

1.2.1 Relation to Knowledge Graphs and Heterogeneous Information Networks

Knowledge graphs [1, 2] and heterogeneous information networks [3] are two closely related concepts to the text-rich networks proposed in this thesis. Knowledge graphs were initially proposed with a strict schema, focusing on (head entity, relation, tail entity) typed triplets. Throught their development over the years, this schema has been relaxed in one way or another. Some modern knowledge graphs [4] also consider textual content. Compared to knowledge graphs, text plays a even more prominent role in our text-rich networks. We focus on the synergy between text and network structure, with text mining applications in mined. While in knowledge graph research, people often put more emphasis on the entities and structures, e.g., learning representations for graph nodes.

Heterogeneous information network is a preceding concept to our text-rich networks. While they sometimes touch on textual content [5], they do not directly work with raw text documents. Instead, research in heterogeneous information networks is usually focused on studying the multi-typed nature of the nodes and edges in the network.

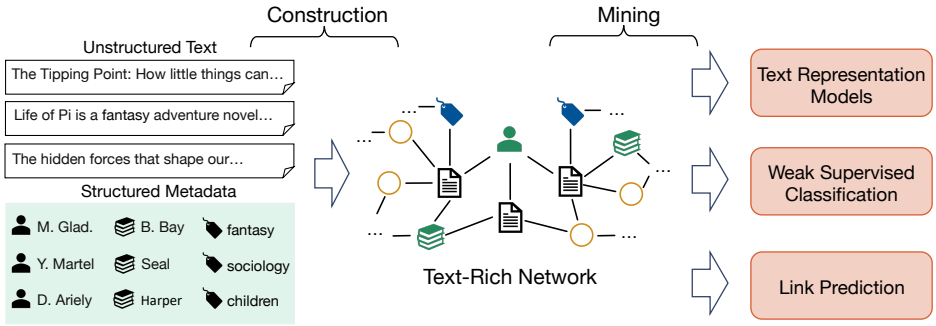


Figure 1.1: The framework of text-rich network construction and mining, with example text derived from the GoodReads book review website.

1.3 METHODOLOGY AND CONTRIBUTIONS

The overview of text mining with text-rich networks is illustrated in Figure 1.1. My research consists of two primary areas of investigation: (1) **Construction and consolidation** of a text-rich network, which explores the process of transforming a semi-structured text corpus into a TRN and extracting structured information from text when metadata is missing or incomplete; and (2) **Mining** of a text-rich network, which investigates weakly supervised text mining tasks using a TRN, with a particular focus on utilizing a TRN for fundamental text representation and specific applications.

1.3.1 Text-Rich Network Construction: Open-World Attribute Mining

Open-world attribute mining plays a crucial role in the construction and consolidation of text-rich networks by enabling the extraction of structured information from unstructured text data with minimal human intervention. In the context of our two main areas of investigation outlined above, open-world attribute mining falls under the first area: text-rich network construction and consolidation. The successful extraction of attributes and values from unstructured text data is essential for building comprehensive and informative text-rich networks that capture the complex relationships between textual content and metadata.

In this work, we focus on product attribute mining as a prominent application context for our open-world attribute mining techniques. Product data presents a pressing need for the open-world setting, as products often have diverse and constantly evolving attributes. To address the challenges of open-world attribute mining, we propose OA-Mine, a principled framework that leverages weak supervision to discover new attribute types and values in the context of product data. The successful application of OA-Mine to open-world product attribute mining contributes to the broader goal of constructing text-rich networks that

seamlessly integrate textual content and structured metadata, enabling the discovery of valuable patterns and insights that may be difficult to uncover using traditional approaches.

Our main contributions in this work are:

- We propose OA-Mine, a principled framework for open-world product attribute mining that discovers both new attribute types and new attribute values with weak supervision, contributing to the construction of comprehensive text-rich networks.
- We introduce an attribute-aware representation fine-tuning framework for pre-trained language models, which leverages weak supervision and abundant unlabeled product text to adapt the language model for distinguishing values from different product attributes, enhancing the quality of the extracted metadata for text-rich network construction.
- We release a human-annotated evaluation dataset for end-to-end evaluation of the open-world product attribute mining task, enabling comprehensive assessment of our proposed framework and facilitating future research in this area.

1.3.2 Text-Rich Network Mining: Language Model Pre-training

In this work, we explore a fundamental aspect of our text-rich network mining framework: language model pre-training that leverages the inherent structure of these networks. By harnessing the power of text-rich networks, we aim to enhance the effectiveness of language model pre-training and improve its performance on downstream tasks.

Language model pre-training has enabled models to capture rich semantic representations from vast amounts of text data. However, the majority of these pre-trained language models (PLMs) are designed for general-domain text corpora and may struggle to accurately understand the nuances and relationships present in text-rich networks. By incorporating the structured information available in these networks, we can enhance the effectiveness of language model pre-training and improve downstream NLP tasks.

Our main contributions in this subsection are:

- We propose a novel approach to language model pre-training that harnesses the power of text-rich networks, demonstrating this approach using Twitter, a platform that naturally exhibits a text-rich network structure through its blend of user-generated content and social engagement logs.

- We introduce a contrastive objective alongside the traditional masked language modeling task by constructing a heterogeneous information network to unify the multi-typed engagement logs and mining socially similar content pairs, allowing our model to capture the semantic connections between documents that may not be apparent from the text alone.

- We demonstrate the power of mining text-rich networks for enhanced language model pre-training, representing a step forward in our mining framework that improves content understanding and has the potential to benefit a wide range of applications across various domains where text-rich networks are prevalent.

1.3.3 Text-Rich Network Mining: Weakly-Supervised Classification

In this work, we explore another crucial application of text-rich network mining: weakly supervised text classification. Text classification is a fundamental task in organizing and understanding semi-structured text corpora, but traditional approaches often rely heavily on extensive human annotation By leveraging the inherent structure of text-rich networks, we aim to develop a novel approach that reduces the reliance on large amounts of labeled data.

We propose LTRN (Learning on Text-Rich Networks), a framework that combines a text analysis module for raw text embedding and classification with a network learning module for semantic-aware propagation of categorical information on the network. By employing co-training and feature sharing, we enable these modules to mutually enhance each other, resulting in improved text classification performance.

Our main contributions in this subsection are:

- We propose LTRN, a joint learning framework on text-rich networks for minimally-supervised text categorization. The framework has two data-driven modules with different inductive biases: a text analysis module and a network learning module. We leverage co-training and feature sharing to train both modules jointly.

- We propose a novel GNN architecture for text-rich network modeling. The proposed model adopts personalized PageRank-based neighborhood sampling and attentive aggregation. The model successfully handles noise on the text-rich network and scales up to large datasets.

- We conduct experiments on two real-world datasets and achieve significant gains over various competitive baselines. On the challenging product categorization dataset with $\sim$700 categories, our model obtains an F-measure of over 90% with only 3 seeds per

category. Compared with a supervised BERT model, our model saves training labels by 20× with only $< 2\%$ sacrifice on F-measure, and it outperforms state-of-the-art self-training and augmentation-based methods trained on the same seed labels by $> 13\%$.

Chapter 2: TEXT-RICH NETWORK CONSTRUCTION: OPEN-WORLD ATTRIBUTE MINING

In the pursuit of constructing comprehensive text-rich networks from semi-structured text corpora, a critical challenge lies in the extraction of structured information from unstructured text data. The interplay between these two components forms the foundation of text-rich networks, enabling the discovery of valuable insights and patterns that may be difficult to uncover using traditional approaches. To fully harness the potential of these networks, it is essential to develop techniques that can effectively mine structured entities, such as attributes and their corresponding values, from the vast amounts of unstructured text data available.

In this chapter, we focus on the task of open-world attribute mining, a crucial step in the construction and consolidation of text-rich networks. Open-world attribute mining addresses the challenge of discovering both new attribute types and their associated values from unstructured text data, with minimal human intervention. This contrasts with conventional information extraction settings in natural language processing, where attribute types are typically known a priori. For example, in named entity recognition, entity types such as person and organization are pre-defined. While this approach may be suitable for certain applications, it can be confining in real-world scenarios where the types of attributes are not known beforehand. We refer to this setting, where both attribute types and values are to be discovered, as the open-world setting.

Our work on open-world attribute mining builds upon the foundation laid out in the introduction chapter, where we outlined the two main areas of investigation in our research: text-rich network construction and mining. In the context of network construction, open-world attribute mining plays a pivotal role in extracting structured information from unstructured text data, enabling us to build more comprehensive and informative text-rich networks. By developing weakly supervised methods that require only a few example values for a small set of known attributes, we aim to reduce the reliance on extensive human annotation or manually curated knowledge bases, making our approach more adaptable to various domains and data types.

We focus on product attribute mining as a prominent application context for our open-world attribute mining techniques. Product data presents a pressing need for the open-world setting, as products often have diverse and constantly evolving attributes. Traditional approaches that rely on pre-defined attribute types may struggle to keep pace with the ever-expanding product catalog. By leveraging data-driven methods in an open-world setting, we can greatly reduce the human effort required to extract relevant attributes and values from product text data. This not only facilitates the construction of comprehensive text-rich networks for

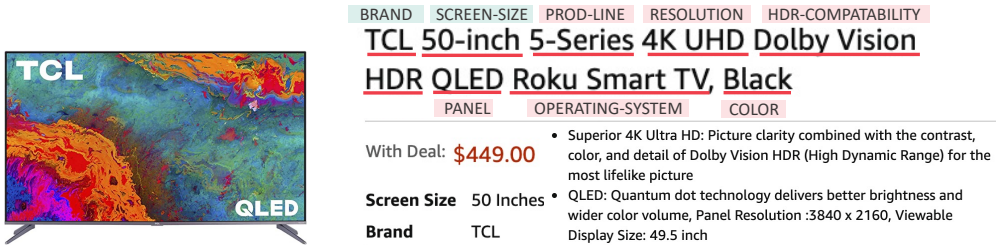


Figure 2.1: An example product title with known attributes annotated in green and new attributes in red. Sellers usually pack important product attribute values in the title for maximum exposure.

product data but also demonstrates the potential of our approach to be applied to other domains where the diversity and evolution of attributes pose similar challenges.

The methods and insights gained from our work on open-world attribute mining contribute to the broader goal of text-rich network construction and consolidation. By integrating the mined attributes and values with the unstructured text data, we can create powerful representations that capture both the textual content and the relationships between entities. These enriched text-rich networks serve as a solid foundation for the subsequent mining tasks, such as weakly supervised text classification and language model pre-training, which we will explore in later chapters. The successful construction of text-rich networks through open-world attribute mining opens up new possibilities for knowledge discovery and data mining, enabling us to uncover valuable patterns and insights that may be difficult to detect using traditional approaches. ¡/intro paragraphs¿

¡overview section¿

2.1 OVERVIEW

Product attributes are essential for online shoppers, as they provide valuable information to help search, recommendation, and product comparison. The massive, constantly changing products bring the open-world challenge. Products we already know may gain new attributes over time – "HDR compatibility" barely comes to mind when people buy a TV ten years ago, but is relevant for many shoppers nowadays. Not to mention, new types of products with distinctively new sets of attributes may emerge in the future. Automatic extraction of product attributes has to meet the open-world challenge. The model must be able to discover both *attribute types* (e.g., brand, referred to as "attributes" hereafter) and *attribute values* (e.g., Samsung, referred to as "values"). Moreover, they must operate with limited supervision, as human annotation can never keep up with the forever-expanding product catalog.

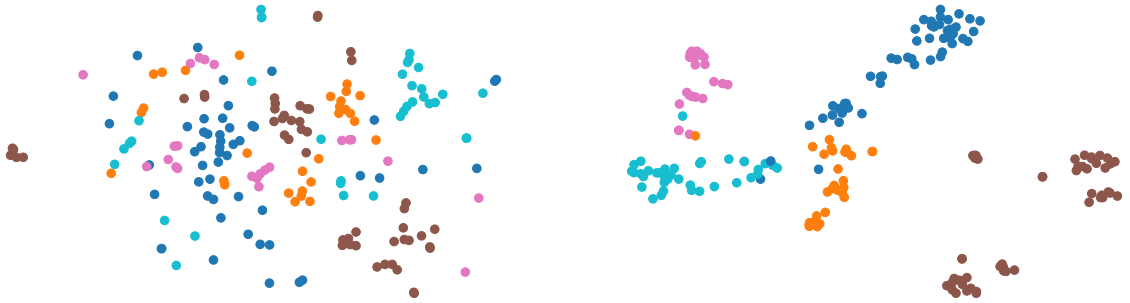
Existing works on attribute mining mostly carry a closed-world assumption on attributes. Pioneer studies such as OpenTag [6] bear "open" in the name, but their open-world assumption is on *values only*. They assume the set of attributes of interest is given as input to the model. The model is trained to find more values for the given attributes, but does not expand to unseen attributes without additional training data. Moreover, they usually rely on extensive human labeling [6] or noisy distantly supervised data [7, 8, 9].

In this work, we study the open-world attribute mining problem with weak supervision. We let the user provide a few example values for a few known attributes. Neither the attributes nor the values given as examples are required to be comprehensive. In addition, we also assume the type of the products are known a priori, as each product type holds a distinct set of applicable attributes (e.g., "resolution" applies to TVs but not shoes), and the attribute mining process should be done with respect to each product type. With limited supervision and abundant product raw text, we aim to discover new attributes and values for products of different types.

Product titles, as the cleanest and most prominent part of product text, possess a number of unique properties. As shown in Figure 2.1, product titles are concise, non-repeating, not written in a way as a general domain natural language. More specifically, we highlight three observations from the data. First, to maximize exposure of products to customers, sellers usually pack the highlights of their product in the title (observation 1: *title first*). Second, a product title rarely contains irrelevant information, and is a collection of attribute values (observation 2: *bag-of-values*). Third, with limited space in the title, the values seldom repeat (observation 3: *value exclusiveness*). For example, if one word describes the resolution of a TV product, the rest of the title will mention other attributes such as screen size. other phrases will cover other attributes of the product. These observations pave the way for a mining-based approach that exploits the concise product titles, is tailored to the language of products, and discovers new attributes and values in an open-world setting.

We propose OAMine, a principled framework for **O**pen-world product **A**ttribute **MIN(E)**ing with weak supervision. Our framework has two main steps, *candidate value generation* and *attribute value grouping*. The first step of our framework is designed based on the *title first* and the *bag-of-values* data observations. It probes an in-domain fine-tuned language model that captures compositional relations between words, and segments product titles into phrases. These phrases are candidates that will either be marked as an attribute value or be excluded as noise in our subsequent step.

Once we obtain the attribute value candidates, our framework finds the major attributes for each product type, and puts the candidates into different attribute groups. We rely on a proper value embedding to propagate supervisions from seed values to other values



(a) BERT trained w/ in-domain corpus (b) After attribute-aware fine-tuning

Figure 2.2: T-SNE visualization of attribute value embedding. Points colored by attribute. After our attribute-aware fine-tuning, embedding is more reflective of attribute type.

when they together form a clique, and also to discover new values when new cliques emerge. We find that pre-trained language model embedding (e.g., BERT embedding) cannot fully capture attribute-specific information, even after MLM [10] pre-training using in-domain corpus, as seen in Figure 2.2. We propose an attribute-aware fine-tuning method, which takes full advantage of *value exclusiveness* of product titles as well as the weak supervision, and optimizes a multitask objective function. After the fine-tuning, embedding becomes more expressive in distinguishing values from different attributes. With well-tuned embedding, we then apply a self-ensemble-based inference method. A density-based clustering component, which can discover open-world new attributes and scrap noise from the candidates, is joint force with a classification component, which improves the model recall by finding more values of the discovered attributes. Our OA-Mine framework improves itself with iterative training, where the discovered attribute types and values help gradually shape the value space for each attribute, as the model becomes more knowledgeable.

We run extensive experiments on over 750K products from 100 product types. In addition to distantly annotated development data, we recruit crowd workers and domain experts to label a gold standard test set for end-to-end evaluation. Our experiments show the superiority of our framework over various state-of-the-art baseline methods. Besides, we also show the model's ability to discover unseen attributes by holding out attributes from training, and demonstrate the model's strong generalization by evaluating on product types with zero supervision.

Our main contributions can be summarized as follows:

- We propose a principled framework for open-world product attribute mining. The framework discovers both new attributes types and new attribute values.
- We propose a novel attribute-aware representation fine-tuning framework for pre-trained

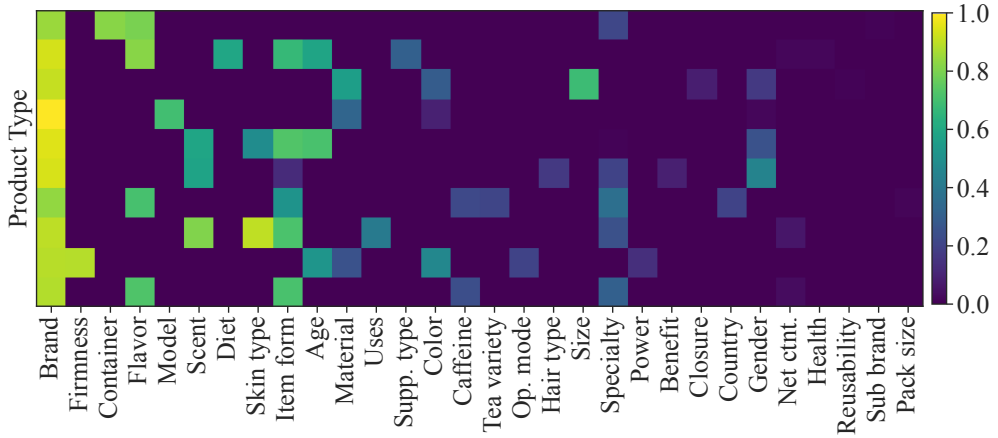


Figure 2.3: Attribute coverage. The cell color at (x, y) is the coverage of attribute x in product type y. 100% means all values of this attribute are found in existing structured information in product profiles.

language models. The framework combines human given weak supervision with abundant unlabeled product text, and adapts the language model to distinguish values from different product attributes.

- We release a human annotated evaluation dataset for end-to-end evaluation of the open-world product attribute mining task.

2.2 PRELIMINARIES

In this section, we formally define the open-world product attribute mining task, and specify our input and output. In addition, we offer some additional insights through data analysis.

Problem Definition. Open-world attribute mining for one type of product t takes a set of products $\mathcal{P}_t$ and aims to discover a set of applicable *attributes* $\mathcal{A}_t$ for those products. Each applicable attribute $A \in \mathcal{A}_t$ is represented by a collection of *attribute values* $A = \{v_1, v_2, ...v_n\}$, which are raw text phrases. The problem is guided by weak supervision, where the user specify a few known values $S_A = \{v_1, v_2, ..., v_k\}$ for a few known attributes $\mathcal{A}_S \subset \mathcal{A}_t$. S_A is called the seed set of attribute A. The seed set size $|S_A|$ and the number of known attributes $|\mathcal{A}_S|$ are small. The predictions are attribute clusters $\bar{\mathcal{A}}_t$ and the goal is to make the predictions $\bar{\mathcal{A}}_t$ close to the ground truth $\mathcal{A}_t$.

The problem is defined for a specific type of product t, because each product type has its unique sets of applicable attributes $\mathcal{A}_t$. Product types are identified by their surface names. Putting all types of products together $\mathcal{P} = \bigcup_{t \in T} \mathcal{P}_t$, the open-world attribute mining aims to

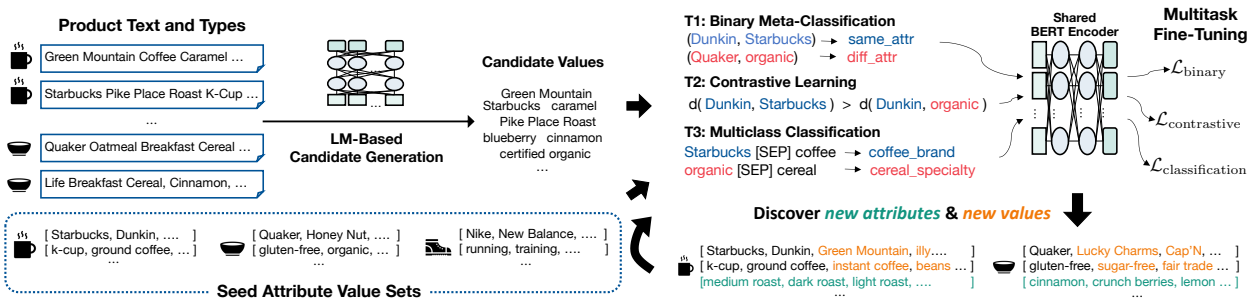


Figure 2.4: Overview of our proposed OA-Mine framework. Starting from product text, we generate candidate attribute values by probing a pre-trained language model (LM). Combined with user given seed sets, we fine-tune the LM with a multitask objective. Finally, we discover new attributes and values with self-ensembled based inference. Predictions are used for training in next framework iteration.

discover all the attributes $\mathcal{A}_t$ for $t \in T$.

In this work, we use product titles from $\mathcal{P}$, and we assume the number of products $|\mathcal{P}|$ is large.

Data Analysis on Attribute Coverage. To show the necessity of open-world product attribute mining, we collected 1,943 product profiles from 10 product types on Amazon.com for data analysis. They contain raw text, as well as structured information, including some attribute values already known. We hired human workers to annotate attribute values mentioned in product titles for a comprehensive analysis. The details of the data collection process can be found in Section 2.6.1.

We would like to know what percentage of attribute values are already presented in the structured information of those online product profiles. In total, the human annotators found 51 distinct attributes from all the products across 10 product types. Of those, only 30 attributes are found in existing structured product profiles. Figure 2.3 shows the percentage of values identified by humans compared to the structured attribute information in product profiles. Rows represent product types, and columns represent attributes. Comparing different rows, we can see different types of products have different applicable attributes. Judged by the color of the cells, lots of values are apparently missing. The data analysis clearly shows the need for open-world attribute mining, as we see missing attributes and values in the existing product profiles.

2.3 FRAMEWORK OVERVIEW

OA-Mine is a two-step framework for weakly supervised open-world product attribute mining (Figure 2.4), with *candidate value generation* followed by *attribute value grouping*. The

first step aims to harvest phrases that are likely attribute values from raw text product titles. Data observations from Section 2.1 show that product titles are collections of attribute values. Our model segments the titles into phrases, essentially breaking down bags-of-attribute-values into individual value candidates. Established phrase mining tools cannot meet our needs, as pre-trained NLP pipelines [11, 12] are not adapted to the product text, while the unsupervised methods [13] typically has assumptions that do not work well on the product titles (shown in Section 2.6.2). We designed a language model probing-based technique, which captures the compositional relations between words for effective phrase segmentation.

Although product titles are primarily bags-of-attribute-values, there are exceptions where things that are not attribute values can be mentioned (e.g., "packaging may vary"). In our first step, we leave them as candidate values and let our second step remove them from the candidate pool. We design our first step to be recall-focused and the second step to be noise-aware, as information lost from the first step cannot be recovered later, while noise can be effectively handled.

Having obtained the attribute value candidates, we find novel attributes and put the candidates into attribute groups in our next step. Our model navigates the embedding feature space of candidate values and discovers dense areas that are likely attribute clusters. Fundamentally, the embedding of candidate values has to be attribute-aware, a requirement we have shown that pre-trained language model embedding cannot satisfy. To achieve this, we draw on the *value exclusiveness* observation from data and combine it effectively with the limited supervision we have from seed values. We optimize a multitask objective, with a binary meta-classification loss to generalize across product types, a contrastive loss to enforce embedding similarity resembles attribute-level similarity, and a classification objective to impose attribute distinctiveness.

Once we have well-trained embedding, a self-ensemble of different parts of the model first discovers new attributes and removes noise by DBSCAN [14] clustering, and then puts more values into attribute groups with classification. Finally, the model improves itself by leveraging the prediction for iterative training.

2.4 CANDIDATE VALUE GENERATION

The first step of our framework aims to generate attribute value candidates from product titles. As product titles are mostly bags-of-attribute-values, we formalize the task as a product title segmentation task. For example, for the product in Figure 2.1, we would like to segment its title into "TCL — 50-inch — 5-Series — 4K UHD —..." Our goal of this step is to include as many value candidates as possible, i.e., it is recall-focused. We allow phrases

that are not attribute values to be kept, but do not want actual attribute values to be lost.

We leverage a pre-trained language model for candidate generation. As observed in prior work [13, 15, 16], pre-trained language models can capture phrases through the attention scores computed by the Transformer model.

We first fine-tune a BERT [10] language model using the Masked Language Model (MLM) [10] objective on the product text. This step is necessary because the product text differs from the general domain natural language text on which BERT is pre-trained. Then we probe the BERT model to generate a score that resembles how likely two words are from the same phrase.

Language Model Probing for Phrase Score. Given a product title as a sequence of words $W = w_1 w_2 ... w_n$, we compute the likelihood $s(w_i, w_{i+1})$ of two adjacent words belonging to the same phrase. We use the following strategy [16]:

$$s(w_i, w_{i+1}) = d\left(\text{BERT}(W/\{w_i\})_i, \text{BERT}(W/\{w_i, w_{i+1}\})_i\right) \tag{2.1}$$

$W/\{\cdot\}$ denotes the original text with words in the set under slash replaced with [MASK]. BERT$(\cdot)$ computes the sequence embedding by applying the BERT model, and $(\cdot)_i$ takes the i-th embedding, i.e., the token embedding of w_i. $d(\cdot, \cdot)$ is a distance function. We use cosine distance in our model.

Phrasal Segmentation. With phrase score $s(w_i, w_{i+1})$ of each adjacent words computed, we can set a threshold for phrasal segmentation. Each pair of words with $s(w_i, w_{i+1})$ above the threshold will be merged into a phrase, and those below the threshold will be segmented into different phrases. We use a small validation set to select the threshold. Empirically we found the model and a single threshold generalize very well across different product types.

One drawback of relying entirely on the language model is that it does not guarantee the completeness of the phrases. For example, "QLED TV" may be combined into one phrase in one product, but separated into two words in another product. This is because the BERT representation is context-dependent, and cannot leverage global frequency signals from the whole corpus. We, therefore, apply frequent pattern mining to combine words that frequently co-occur in the corpus, which further improves the completeness of the attribute value candidates.

2.5 ATTRIBUTE VALUE GROUPING

Once we have generated candidate values from product titles, we are ready to group them into attribute clusters. Given the open-world challenge in our problem setting, standard token classification approaches cannot suffice. Fully unsupervised clustering methods also fall short, as they rely on the pre-trained representation of phrases, which is not attribute-aware (Figure 2.2).

We identify two main requirements for discovering new attributes and values: (1) The candidate value representation must be attribute-aware, placing values from the same attribute close to each other, and those from different attributes far apart; (2) The model must generalize to unseen attributes and to various product types. Bearing these two requirements in mind, we first design a multitask fine-tuning approach that focuses on attribute awareness and generalization. Then we apply a self-ensemble inference to discover new attributes and values.

2.5.1 Attribute-Aware Representation Fine-Tuning

Given the candidate attribute values, we use a shared encoder to obtain their embedding, and then we apply three different objective functions to fine-tune the model.

Attribute Value Encoder. Given a candidate attribute value v and its textual context in the product title W, we use a BERT [10] encoder to generate the token representation of the whole sequence. Next, we apply an average pooling on the token representations of the attribute value to obtain its representation $\boldsymbol{h}_v$. Specifically,

$$\hat{\boldsymbol{H}} = \left[\hat{\boldsymbol{h}}_1 \hat{\boldsymbol{h}}_2 ... \hat{\boldsymbol{h}}_n\right] = \text{BERT}(W) \tag{2.2}$$

$$\boldsymbol{h}_v = \sum\nolimits_w \left(\hat{\boldsymbol{h}}_w \cdot \mathbb{1}\left[w \in v\right]\right) / \sum\nolimits_w \mathbb{1}\left[w \in v\right] \tag{2.3}$$

The characteristic function $\mathbb{1}\left[\cdot\right]$ equals to 1 for words in the candidate value, and 0 for the words in the context, thus masking out the hidden representation from the context.

Recall that our input seed sets are collections of attribute values without context. When encoding the values from the seed sets, we string match them to their occurrences in the products, and use those occurrences for their contextualized representation.

Binary Meta-Classification. The ultimate goal of fine-tuning is to embed values from the same attribute close to each other and embed values from different attributes far apart. We build a binary classification model f to directly optimize for this goal: given a pair of values v_1, v_2, let $f(v_1, v_2)$ be 1 if v_1, v_2 are from the same attribute, and -1 otherwise.

We use a BERT bi-encoder [17] with cosine similarity objective as the model f.

$$f(u, v) = \text{cosine\_sim}\left(\boldsymbol{h}_u, \boldsymbol{h}_v\right) \tag{2.4}$$

$$\mathcal{L}_{\text{binary}} = \sum_{(u,v)\in P} \left\|1 - f(u, v)\right\|^2 + \sum_{(u,v)\in N} \left\|-1 - f(u, v)\right\|^2 \tag{2.5}$$

P and N are positive and negative example sets respectively. We choose BERT bi-encoder as opposed to the standard BERT cross-encoder [10] because the bi-encoder is more efficient at inference time, as we will see in Section 2.5.2.

Note that we would like the model to capture the attribute distinctiveness, not just on one attribute of one product type. Rather, we would like to build a *meta-classifier* which generalizes to all attributes and all product types. We carefully sample the positive training pairs P and the negative training pairs N from both the seed sets and the unlabeled corpus. We generate positive pairs from values in the same attribute seed set, and negative pairs from different seed sets. In addition, based on the *value exclusiveness* data observation, we generate strong negative training data by first sampling a product title from the corpus, and then sampling a pair of values from the same title. Our method generates higher quality pairs compared to random negative sampling.

Contrastive Learning. In addition to directly optimizing the binary meta-classification loss, we find it beneficial to bind together the positive and the negative training pairs with a contrastive learning loss [18, 19, 20, 21]. The contrastive learning objective takes a triplet of $(v_{\text{anc}}, v_{\text{pos}}, v_{\text{neg}})$, and enforce the embedding distance from the anchor v_{anc} to the positive example v_{pos} to be smaller than that from the anchor v_{anc} to the negative example v_{neg}.

Contrastive learning has different instantiations in the choice of the loss function. Recent studies usually explicitly generate positive pairs with data augmentation but implicitly generate negative pairs [21]. Since we can explicitly generate strong negative pairs, we resort to a triplet loss function [18]:

$$\mathcal{L}_{\text{contrastive}} = \sum_{(v_a, v_p, v_n)} \max\left(\left\|f(v_a, v_p)\right\|^2 - \left\|f(v_a, v_n)\right\|^2 + \alpha, 0\right) \tag{2.6}$$

α is a constant margin value. The triplets are generated by binding our positive pairs P and negative pairs N.

Multi-class Classification. The binary and the contrastive loss functions would shape the hidden feature space to be attribute-aware, as they manipulate the embedding space, so values that fall into the same attribute are pulled closer while values that belong to different types are pushed away from each other. However, neither of those loss functions enforce

the class distinctiveness of attribute values. Adding class distinctiveness by a multi-class classification loss would push the embedding from different attributes further apart. It offers additional benefits at inference time, shown in Section 2.5.2, where it would improve the recall of the model.

Let us assume we have already discovered the complete set of attributes. Now, we can build a multi-class classifier to put each value into an attribute.

The challenge is scaling up to multiple product types, as each product type has a distinct set of applicable attributes. For example, the flavor of coffee can be very different from that of cereal. Most prior works do not distinguish product types and are suboptimal.

We want to avoid building a separate classifier for each product type, while maintaining the product type awareness of the model. As such, we feed both the attribute value with context and the product type surface name to the multi-class classification model, delimited by the [SEP] special token. The output label space is the union of applicable attributes for all product types, e.g., "coffee_flavor", "tea_brand". We use the cross-entropy loss function for classification.

$$\hat{\boldsymbol{y}} = \text{Softmax}(\text{Linear}(\text{BERT}(W[\text{SEP}]t))) \tag{2.7}$$

$$\mathcal{L}_{\text{classification}} = \text{CrossEntropy}(\hat{\boldsymbol{y}}, \boldsymbol{y}) \tag{2.8}$$

W is the attribute value v in its context, t is the product type surface name, and $\boldsymbol{y}$ is the one-hot encoding of the class label.

Note that we only use the attributes with seed values as our classes at the beginning of the training. We expand to more attributes through iterative training, where the predictions from the last iteration will reveal the complete attribute landscape.

2.5.2 Self-Ensemble Based Inference

Once the model has been fine-tuned, we are ready to run an open-world inference to discover new attributes and values, given the attribute-aware embedding for each value.

Attribute Discovery with DBSCAN. Thanks to the bi-encoder training with binary and contrastive objectives, the similarity of value embedding now reflects the likelihood of two values coming from the same attribute.

We opt for DBSCAN [14] to discover new attributes as it has several desirable properties. First, its local density requirement consolidates the attribute-level similarity computed by pairwise distance. Second, it discovers new attributes by finding dense cliques in the embedding space. Third, it excludes values that it deems noise if the embedding of a value is

far from any dense clusters.

Improving Recall with Classifier Inference. We run the multi-class classifier component after the fine-tuning. Each occurrence of a candidate attribute value is classified into an attribute cluster previously discovered by the DBSCAN model. The corpus level prediction of an attribute value is generated by majority voting of each occurrence of that value.

Ensembling both Modules. Empirically, we have observed that the DBSCAN inference is very precision focused, leaving a large number of values out in a "noise" cluster. Whereas the classifier inference is recall-driven, expanding the model's coverage effectively. We, therefore, combine the attribute discovery ability of the DBSCAN inference with the classifier. Once DBSCAN discovered attribute clusters with high precision, we run the classifier on the noise cluster to include more values that were mistakenly excluded from the prediction.

2.5.3 Iterative Training

After one iteration of the framework, we have discovered new attributes and values. They are clusters of attribute values for different product types, and have the same format as our input seed sets. Therefore, we leverage the confident predictions as training data in the next iteration to boost the model performance further.

There are two key benefits of iterative training. First, it augments the existing labeled seed sets, expanding the minimal supervision. Second, it offers a more complete landscape of attributes in the dataset. In our open-world setting, we do not have comprehensive supervision for all attributes in the beginning. The model predictions will alleviate this issue, especially for the classification objective in our framework, which operates on a set of known attributes. The complete training algorithm is shown in Algorithm 2.1.

2.6 EXPERIMENTS

2.6.1 Datasets

Training Data. We collect publicly available product profile data from Amazon.com for 100 different product types. Besides product titles, which will be used as textual descriptions of products, we also collect structured attributes found in these profiles (see Figure 2.1 as an example). For each product type, we find its applicable attributes, e.g., "brand", "roast type" for coffee and "gender", "sports type" for shoes. For each applicable attribute, we examine the values mentioned in the product profiles that we collected, and take the most frequent

Algorithm 2.1: Attribute Value Grouping of OA-Mine

Input: Seed sets S for all product types T, candidate attribute values C, pre-trained model Θ.

repeat
  `// Attribute-Aware Representation Fine-Tuning`
  Initialize empty sets $D_{\text{binary}}, D_{\text{contrastive}}, D_{\text{classification}}$;
  Initialize empty prediction set P ;
  for $t \in T$ **do**
    $D_t \leftarrow \text{generate\_data}(S_t \cup P_t, C_t)$; `// Sec 2.5.1`
    Expand $D_{\text{binary}}, D_{\text{contrastive}}, D_{\text{classification}}$ with D_t ;
  $D \leftarrow D_{\text{binary}} \cup D_{\text{contrastive}} \cup D_{\text{classification}}$;
  Shuffle and batch D ;
  for $b \in D$ **do**
    Compute loss $L(\Theta)$
      $L(\Theta)$ = Eq. 2.5 for binary meta-classification
      $L(\Theta)$ = Eq. 2.6 for contrastive learning
      $L(\Theta)$ = Eq. 2.8 for classification
    Update model $\Theta \leftarrow \Theta - \epsilon\nabla(\Theta)$;
  `// Self-Ensemble Inference`
  $P \leftarrow \{\}$; `// Empty previous predictions`
  for $t \in T$ **do**
    $P_t \leftarrow \text{run\_inference}(C_t, \Theta)$; `// Sec 2.5.2`
    $P \leftarrow P \cup P_t$;
until *max_iter*;

values as the seed set for that attribute. We keep up to 5 seed values for each attribute.

Test Label Set. For end-to-end evaluation purposes, we manually selected products from 10 product types, sampled up to 200 products for each product type (1,943 products in total), and hired human workers for labeling. We created a labeling tool that allows human workers to highlight attribute values mentioned in product titles and label their type. We defined a few applicable attributes we already know for each product type and added an option for human workers to label new attributes not seen in the predefined set.

First, each product is labeled by 5 crowd workers on Amazon Mechanical Turk. Then, the labeled profiles are reviewed by a domain expert for consolidation. After the values for known attributes are consolidated, we review the dataset for another pass and assign proper attribute types to those previously labeled as "new attribute". On average, there are 11.5 attributes per product type marked by humans, and each attribute has 48.1 unique values.

The labels are gathered from human annotations, and are clusters of attribute values for the 10 selected product types. For example, labeled attribute values for coffee products are {Brand: [Starbucks, Dunkin, ...], Roast Type: [medium roast, dark roast, ...], ...}.

Table 2.1: Evaluation on Attribute Value Candidate Generation. Methods are divided into pre-trained, distantly supervised, and unsupervised, from top to bottom.

Methods	Entity-Prec.	Entity-Rec.	Entity-F1	Corpus-Rec.
spaCy [11]	31.19	19.15	23.73	50.02
FlairNLP [12]	34.81	24.33	28.64	52.17
AutoPhrase [22]	26.58	29.67	28.04	32.39
UCPhrase [13]	35.01	19.66	25.18	37.50
OA-Mine	**42.53**	**53.29**	**47.30**	**64.10**

Development Label Set. In addition to the human-annotated test set on selected 10 attribute types, we create a more extensive development set covering all 100 product types. The labels are generated from the existing attribute information in the collected product profiles, using the same procedure described in the "Training Data" section for seed sets curation. Aligned with training seed sets, we only keep values in the development when they are discovered by our candidate generation step. On average, each attribute has 29.7 unique values.

With a limited budget, the development set would allow us to evaluate the performance of different models on 100 product types. Note that for most prior works on product attribute mining [7, 8, 9], the authors use the same method for gold-standard evaluation. While in this paper, the development set serves the purpose of relative performance comparison.

Reproducibility. Our code and data can be found at https://www.github.com/xinyangz/OAMine.[1]

2.6.2 Candidate Generation Performance

We compare the attribute value candidate generation module of our framework to the following baseline methods on the human-annotated test set:

- **spaCy** [11] is an industrial library with a pre-trained statistical noun phrase chunking model.
- **FlairNLP** [12] is a neural NLP library. We use its phrase chunking model [2] that is based on a neural sequence tagger.
- **AutoPhrase** [22] is a distantly-supervised phrase mining tool. It leverages a high quality

[1] Note that although there were a few existing works [6, 7, 8, 9] on product attribute mining, only one of them [7] provided a subset of their experiment data, and the dataset was labeled with distant supervision, thus do not serve our needs.

[2] https://huggingface.co/flair/chunk-english

Table 2.2: End-to-end evaluation on development data. Results are average of 3 runs. Bold faced numbers indicate statistically significant results from t-test with 99% confidence.

		Dev Set (100 product types)			
Method Type	**Method**	ARI	Jaccard	NMI	Recall
Sequence tagging (closed-world)	BiLSTM-Tag	0.299	0.354	0.422	0.565
	OpenTag [6]	0.244	0.324	0.334	0.593
	SU-OpenTag [7]	0.637	0.598	0.607	0.525
Unsupervised clustering	BERT+AG-Clus	0.249	0.446	0.585	0.742
	BERT+DBSCAN	0.133	0.146	0.507	0.131
Weakly sup. clustering	DeepAlign+ [23]	0.175	0.226	0.336	0.729
	OA-Mine (no multitask)	0.671	0.634	0.610	0.458
	OA-Mine	**0.704**	**0.689**	**0.629**	0.747

Table 2.3: End-to-end evaluation on test data. Results are average of 3 runs. Bold faced numbers indicate statistically significant results from t-test with 99% confidence.

		Test Set (10 product types)			
Method Type	**Method**	ARI	Jaccard	NMI	Recall
Sequence tagging (closed-world)	BiLSTM-Tag	0.175	0.219	0.374	0.162
	OpenTag [6]	0.160	0.247	0.357	0.165
	SU-OpenTag [7]	0.411	0.340	0.542	0.162
Unsupervised clustering	BERT+AG-Clus	0.386	0.308	0.504	**0.430**
	BERT+DBSCAN	0.385	0.412	0.575	0.186
Weakly sup. clustering	DeepAlign+ [23]	0.257	0.208	0.426	0.389
	OA-Mine (no multitask)	0.601	0.518	0.733	0.225
	OA-Mine	**0.712**	**0.650**	**0.781**	0.275

phrase dictionary from Wikipedia and trains a statistical classifier based on POS-tags and statistical features of text spans.

- **UCPhrase [13]** is an unsupervised phrase mining framework. It first finds "core phrases" by mining frequent closed sequential patterns. It then uses the core phrases to train a CNN-based phrase classifier which leverage features from attention maps extracted from pre-trained language models.

Evaluation Metrics. We report micro-averaged entity level precision, recall, F1, which are computed per instance. In addition, we also report corpus-level recall where the predictions on all products are compared to the labeled set.

Results. As shown in Table 2.1, our model consistently outperform the baseline methods by a large margin. spaCy and FlairNLP are sub-optimal on our dataset because they are pre-trained on general domain corpora, with very different language characteristics than

product titles. AutoPhrase performs poorly as it lacks in domain distant training dictionary. UCPhrase struggles on our dataset because (1) it can only capture multi-word phrases while attribute values can be single tokens, (2) the core phrases generated by pattern mining are low quality. Unlike long documents on which the authors designed their model, product text is noisy and repetitive, undermining the pattern mining method.

2.6.3 End-to-End Evaluation

The end-to-end evaluation runs each model on the complete training set, and compares their performance on the dev and test sets.

Note that there is no existing work that solves the open-world attribute mining problem to our knowledge. We include several sequence tagging-based models, which are the de-facto models for closed-world product attribute mining. We also combine the candidate generation results from our model with a few open-world classification and clustering models for evaluation.

- **BiLSTM-Tag** is a sequence tagging model with a bi-directional LSTM encoder. The model trains a stand alone tagging model for each known attribute in the seed set.
- **OpenTag** [7] improves the BiLSTM-Tag model by adding an attention layer. The original model adopts a CRF layer for prediction. However, we found that the CRF layer consistently under-performs a linear layer in our weakly supervised setting. Therefore, we use a linear prediction layer instead.
- **SU-OpenTag**[7][3] is an end-to-end product attribute mining framework. It encodes product text and attribute surface name separately with BERT [10], and combines them with LSTM and attention layers. Unlike OpenTag, the model trains a single tagger for all the known attributes. We substitute the CRF layer in the original paper with a linear layer as it performs better.
- **BERT+AG-Clus** leverage BERT embedding for agglomerative hierarchical clustering. We use the attribute value candidates generated by our framework as the input to the clustering method.
- **BERT+DBSCAN** leverage BERT embedding for DBSCAN density-based clustering. We use the attribute value candidates generated by our framework as the input to the clustering method.
- **DeepAlign+** [23] is an open-world intent classification framework with clustering and self training. We adapt it for open-world token classification.

[3] A more recent work [8] improves the model by innovating on the CRF layer. However, CRF does not perform well in our weakly supervised setting. Therefore, we exclude [8] from comparison.

- **OA-Mine (no multitask)** is an ablation of our model. It uses binary meta-classification as its only fine-tuning objective function. Meanwhile, it relies on DBSCAN only for inference, as the classifier component is turned off during training.

Experiment Setting. For sequence tagging-based models that use distant supervision, following [7], we string match our weak supervision sets to the product text to obtain the annotated corpus. The clustering models and the DeepAlign+ model can only handle open-world classification, but not value extraction. As such, we use the candidate values generated by our model as the input to those models. After model training, we run all the methods on the whole corpus for inference, and calculate the evaluation metrics on the product types from the development and test sets. Therefore, our evaluation is transductive. All models that utilize BERT use the same pre-trained model with in-domain MLM fine-tuning.

The main parameters of our model are candidate generation threshold, DBSCAN parameters, and number of iterations. We set the candidate generation threshold as described in Section 2.4. The DBSCAN parameters are selected after our attribute-aware fine-tuning. When generating fine-tuning data, a small validation set was held out. The model's distance threshold for binary meta-classification performs well for DBSCAN, as they all rely on pairwise cosine distances. We run iterative training until stable – up to 5 iterations in practice.

Evaluation Metrics. We compare the predicted attribute clusters with the ground truth clusters derived from human annotations. We use standard clustering evaluation metrics, namely **adjusted Rand index (ARI)**, **Jaccard**, **normalized mutual information (NMI)** for clustering quality evaluation, and **recall** for clustering coverage evaluation. We refer readers to [24, Chapter 17] for details of these metrics. All metrics except for recall are *computed on labeled attribute values* and micro-averaged across attributes and product types. The range of ARI is -1 to 1 and all the other metrics are 0 to 1, larger is better.

Results. As shown in Table 2.2 and 2.3, our model achieves the best performance across the board with the exception of recall on the test set. Recall of BERT+AG-Clus and DeepAlign+ are superior because they cannot remove noise from the candidates, and include everything in their predictions. We can clearly see the downsides of that, as the rest of the metrics indicating the quality of the predicted clusters are significantly lower than our model.

Sequence tagging-based models perform poorly for two main reasons. First, they require more training data to work, and second, they cannot discover new attributes. SU-OpenTag works better than the other two as it leverages BERT pre-trained embeddings.

Comparing our model to the BERT+DBSCAN model, we clearly see the necessity of attribute-aware fine-tuning. In fact, without fine-tuning, DBSCAN is extremely sensitive to

Table 2.4: Performance on discovering new attributes. Experiment conducted with 5-fold cross-validation, where each fold holds out 20% attributes from training.

Methods	ARI	Jaccard	NMI	Recall
BERT+AG-Clus	0.215	0.372	0.308	0.832
BERT+DBSCAN	0.199	0.431	0.129	0.370
DeepAlign+	0.192	0.329	0.303	0.831
OA-Mine	0.599	0.743	0.489	0.688

Table 2.5: Comparing model predictions on unseen attributes during cross-validation. Red is error.

Attribute	**Method**	**Predicted Cluster**
Coffee Brand	BERT+AG-Clus	green mountain, folgers, coffee fool, maxwell house, coffee roasters, nescafe, eight o clock, ...
	DeepAlign+	gourmet, keurig brewers, starbucks, green mountain coffee, donut, dunkin donuts, ...
	OA-Mine	starbucks, green mountain, folgers, coffee fool, maxwell house, nescafe, san marco coffee, ...
Laundry Detergent Form	BERT+AG-Clus	powder, bottle, pacs, original, 2, pods, 32 loads, ...
	DeepAlign+	liquid, laundry, wash, pack, stain, natural, ...
	OA-Mine	liquid, powder, bottle, spray, carton, pods, soap, ...

parameters and performs poorly despite our tremendous effort in tuning it to work.

The DeepAlign+ method is surprisingly worse than the AG-Clus unsupervised clustering in many metrics. This is because the initialization that it based on is poor on our data, and the adaptive clustering component, i.e., self-training, fails to work on the poor initialization. Moreover, the BERT representation used by DeepAlign+ is not attribute-aware.

Comparing our model to the ablation without multitask objective and self-ensemble inference, we can see performance increase across the board, with both better quality and better recall in our main model. This demonstrates that multitask training boosts the attribute-awareness of the model, leading to better quality, while the self-ensemble generated better coverage.

2.6.4 Evaluation on New Attributes

To test model's ability on discovering new attributes, we run an experiment on the development set, as we have aligned attribute types for training seed sets and evaluation label sets. For all the available seed sets in all product types, we hold out 20% attribute types, and use the rest for training. After training and inference, we test the model's performance on the 20% held-out attributes. We report the performance from 5-fold cross-validation on held out attributes. We exclude the sequence labeling-based methods, as they are closed-world and cannot discover new attributes.

The results are shown in Table 2.4. We observe that our method generates significantly higher quality clusters measured by ARI, Jaccard, and NMI scores. Note that BERT+AG-Clus and DeepAlign+ cannot exclude noise from the predictions, resulting in lower quality, albeit high recall.

Table 2.5 shows a case study of models' ability in discovering unseen attributes. The selected attributes are held out during training. The cluster that most resembles the target attribute is selected for each method. Values are sorted by frequency. We omit BERT+DBSCAN as it consistently under-performs our model. As we can see, our model generates the most accurate and semantically coherent clusters. BERT+AG-Clus and DeepAlign+ cannot remove noise from predictions, resulting in lower quality. DeepAlign+ also suffers from semantic drift from adaptive clustering.

2.6.5 Evaluation on New Product Types

We test the ability of our model to generalize to product types with no weak supervision seed sets. In this experiment, we exclude the seed sets for product types in the test set, and use the remaining seed sets from the rest of the 90 product types for training. The model has full access to the raw product text. After training, we run inference on the products not seen during training, and compute the performance on the test set. None of the existing attribute mining or open-world classification baselines can handle this challenging experiment setting. We compare to the unsupervised clustering methods and also to our main experiment result.

The results are seen in Table 2.6. Compared to the unsupervised clustering, our model generates significantly higher quality predictions, as measured by ARI, Jaccard, and NMI. Without access to supervision for these product types, our model performs understandably slightly worse than the full access model reported in the main experiment Table 2.2 and 2.3. The difference, however, is not large, showing the encouraging generalization of our model.

Table 2.6: Performance on new product types. Models tested on product types not seen during training.

Methods	ARI	Jaccard	NMI	Recall
BERT+AG-Clus	0.386	0.308	0.504	0.430
BERT+DBSCAN	0.385	0.412	0.575	0.186
OA-Mine	0.658	0.609	0.702	0.231

2.7 RELATED WORK

Product Attribute Mining. Most prior work on product attribute mining formalize the problem as a sequence labeling task. Inspired by named entity recognition models, earlier work leverage statistical models [25] for extraction. With the advancement of deep learning, the most prominent systems designed in recent years adopt BiLSTM-CRF [6, 8] or BERT-BiLSTM-CRF [7] architectures for attribute value extraction. Supervised method combined with active learning [6] was explored in OpenTag [6], while follow up works typically settle on distant supervision [7, 8, 9]. Besides sequence labeling-based methods, AVEQA [9] explored question answering-based models for attribute value extraction.

The major different between our work and the existing works is that we approach the problem with open-world assumptions on both attributes and values, while prior works assume the attributes of interest are given as input. For supervision, we adopt weak supervision, which is a small amount of high quality data, instead of distant supervision, where a large amount of noisy data is used.

Open-World Text Classification. Although few work studies entity extraction in the open-world setting, a few prior works have explored open-world text classification with different application scenarios. Xu et al. [26] proposed a model for open-world product classification, where they learn a meta-classifier to either match a new example to a seen class or to assign it to a new class. Zhang et al. [23] studies open-intent discovery, which is essentially open-world text classification, with an adaptive clustering and self-training-based approach. Most of these work relies on pre-trained language representation and adaptive clustering [27].

Our problem setting is more challenging than open-world text classification, as we have to extract attribute values from raw text. In terms of methodology, our proposed model can take advantage of unique characteristics of the product text, and can learn better representation. As seen in our experiments, the adaptive clustering-based methods do not work well out-of-the-box, as the product text is noisy and leads to poor convergence of

adaptive clustering.

2.8 CONCLUSIONS AND FUTURE WORK

In this work, we proposed a principled framework for open-world product attribute mining. The framework includes a novel candidate generation method, an attribute-aware representation fine-tuning model, and achieves open-world inference through self-ensemble. Our framework offers strong performance and generalize to unseen attributes and product types. In the future, we will extend our model to handle additional text fields besides product titles, such as using product introductions. Improving the classifier to make open-world inference is also a promising direction.

We shall note that, by leveraging the data observations introduced in section 2.1, we are imposing inductive bias to the model and to our resulting corpora. In other words, the proposed method would have a limited scope to only work on consolidating text-rich networks where the textual content follow those data assumptions. As our future work, with the more powerful pre-trained language models emerging recently, we could revise the candidate generation stage of the framework so that it would not make any assumptions on the textual content, and would better fit into our overall vision of consolidating text-rich networks.

Chapter 3: TEXT-RICH NETWORK MINING: LANGUAGE MODEL PRE-TRAINING

Building upon the foundation of text-rich network construction discussed in the previous chapter, we now transition to the exploration of text-rich network mining. In this chapter, we focus on a fundamental aspect of our mining framework: language model pre-training that leverages the inherent structure of these networks. By harnessing the power of text-rich networks, we aim to enhance the effectiveness of language model pre-training and improve downstream natural language processing (NLP) tasks.

Language model pre-training has revolutionized the field of NLP, enabling models to capture rich semantic representations from vast amounts of text data. However, the majority of these pre-trained language models (PLMs) are designed for general-domain text corpora and may struggle to understand the nuances and relationships present in text-rich networks accurately. By incorporating the structured information available in these networks, we can enhance the effectiveness of language model pre-training and improve downstream NLP tasks.

In this work, we propose a novel approach to language model pre-training that harnesses the power of text-rich networks. We demonstrate this approach using Twitter, a platform that naturally exhibits a text-rich network structure through its blend of user-generated content and social engagement logs. By constructing a heterogeneous information network to unify the multi-typed engagement logs, we mine socially similar content pairs and introduce a contrastive objective alongside the traditional masked language modeling task. This innovative approach allows our model to capture the semantic connections between documents that may not be apparent from the text alone.

The development of this text-rich network-driven language model pre-training represents a step forward in our mining framework. By leveraging the rich structural information embedded in these networks, we demonstrate the power of mining text-rich networks for enhanced language model pre-training. This approach improves content understanding and has the potential to benefit a wide range of applications across various domains where text-rich networks are prevalent.

3.1 OVERVIEW

The proliferation of pre-trained language models (PLMs) [10, 28] based on the Transformer architecture [29] has pushed state-of-the-art across many tasks in natural language processing (NLP). As an application of transfer learning, these models are typically trained on massive text corpora and, when fine-tuned on downstream tasks, have demonstrated state-of-the-art

Figure 3.1: (a) This mock-up shows a short-text Tweet and social engagements such as Faves, Retweets, Replies, Follows that create a social context to Tweets and signify Tweet appeal to engaging users. (b) Co-engagement is a strong indicator of Tweet similarity.

performance.

Despite the success of PLMs in general-domain NLP, fewer attempts have been made in language model pre-training for user-generated text on social media. In this work, we pre-train a language model for Twitter – a prominent social media platform where users post short messages called Tweets. Tweets contain informal diction, abbreviations, emojis, and topical tokens such as hashtags. As a result, PLMs designed for general text corpora may struggle to understand Tweet semantics accurately. Existing works [30, 31] on Twitter LM pre-training do not address these challenges and simply replicate general domain pre-training on Twitter corpora.

A distinctive feature of Twitter social media is the user interactions through Tweet engagements. As seen in Figure 3.1, when a user visits Twitter, in addition to posting Tweets, they can perform a variety of *social* actions such as "Favoriting", "Replying" and "Retweeting" Tweets. The wealth of such engagement information is invaluable to Tweet content understanding. For example, the post "bottom of the ninth, two outs, and down by one!!" would be connected to baseball topics by its co-engaged Tweets, such as "three strikes and you're out!!!". Without the social contexts, a conventional text-only PLM objective would struggle to build this connection. As an additional benefit, a socially-enriched language model will also vastly benefit common applications on social media, such as social recommendations [32] and information diffusion prediction [33, 34].

We introduce TwHIN-BERT– a multilingual language model for Twitter pre-trained with

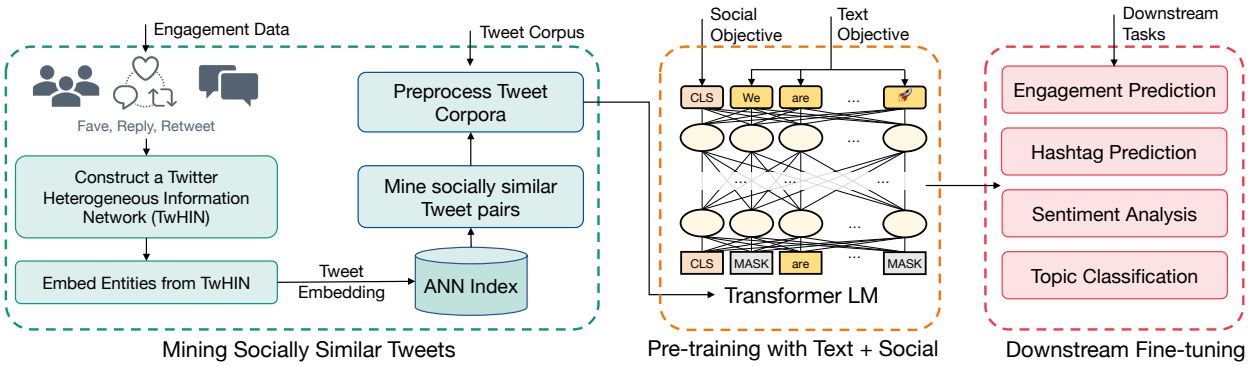


Figure 3.2: We outline the end-to-end TwHIN-BERT process. This three-step process involves (1) mining socially similar Tweet pairs by embedding a Twitter Heterogeneous Information Network (2) training TwHIN-BERT using a joint social and MLM objective and finally (3) fine-tuning TwHIN-BERT on downstream tasks.

social engagements. The key idea of our method is to leverage *socially similar Tweets* for pre-training. Building on this idea, TwHIN-BERT has the following features. (1) We construct a Twitter Heterogeneous Information Network(TwHIN) [35] to unify the multi-typed user engagement logs. Then, we run scalable embedding and approximate nearest neighbor search to sift through hundreds of billions of engagement records and mine socially similar Tweet pairs. (2) In conjunction with masked language modeling, we introduce a contrastive social objective that enforces the model to tell if a pair of Tweets are socially similar or not. Our model is trained on 7 billion Tweets from over 100 distinct languages, of which 1 billion have social engagement logs.

We evaluate the TwHIN-BERT model on both social recommendation and semantic understanding downstream evaluation tasks. To comprehensively evaluate on many languages, we curate two large-scale datasets, a social engagement prediction dataset focused on social aspects and a hashtag prediction dataset focused on language aspects. In addition to these two curated datasets, we also evaluate on established benchmark datasets to draw direct comparisons to other available pre-trained language models. TwHIN-BERT achieves state-of-the-art performance in our evaluations with a major advantage in the social tasks.

In summary, our contributions are as follows:

- We build the first ever socially-enriched pre-trained language model for noisy user-generated text on Twitter.
- Our model is the strongest multilingual Twitter PLM so far, covering 100 distinct languages.
- Our model has a major advantage in capturing the social appeal of Tweets.
- We open-source TwHIN-BERT as well as two new Tweet benchmark datasets: (1) hashtag prediction and (2) social engagement prediction.

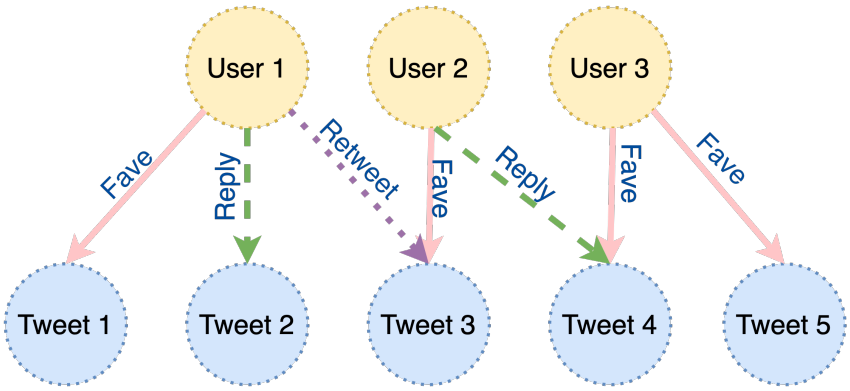


Figure 3.3: Twitter Heterogeneous Information Network (TwHIN) capturing social engagements between users and Tweets.

3.2 TWHIN-BERT

In this section, we outline how we construct training examples for our social pre-training objectives and subsequently train TwHIN-BERT with social and text objectives. As seen in Figure 3.2, we first construct and embed a user-Tweet engagement network. The resultant Tweet embeddings are then used to mine pairs of socially similar Tweets. These Tweet pairs and others are used to pre-train TwHIN-BERT, which can then be fine-tuned for various downstream tasks.

3.2.1 Mining Socially Similar Tweets

With abundant social engagement logs, we (informally) define socially similar Tweets as *Tweets that are co-engaged by a similar set of users.* The challenge lies in how to implement this social similarity by (1) fusing heterogeneous engagement types, such as "Favorite", "Reply", "Retweet", and (2) efficiently mining billions of similar Tweet pairs.

To address these challenges, TwHIN-BERT first constructs a **Tw**itter **H**eterogeneous **I**nformation **N**etwork (TwHIN) from the engagement logs, then runs a scalable heterogeneous network embedding method to capture co-engagement and map Tweets and users into a vector space. With this, social similarity translates to embedding space similarity. Subsequently, we mine similar Tweet pairs via ANN search on the Tweet embeddings.

Constructing TwHIN. We define and construct TwHIN as:

Definition 3.1 (TwHIN)**.** *Our Twitter Heterogeneous Information Network is a directed bipartite graph* $G = (U, T, E, \phi)$*, where* U *is the set of user nodes,* T *is the set of Tweet nodes,*

$E = U \times T$ is the set of engagement edges. $\phi : E \mapsto \mathcal{R}$ is an edge type mapping function. Each edge $e \in E$ belongs to a type of engagement in $\mathcal{R}$.

Our curated TwHIN (Figure 3.3) consists of approximately 200 million distinct users, 1 billion Tweets, and over 100 billion edges. We posit that our TwHIN encodes not only user preferences but also Tweet social appeal. We perform scalable network embedding to derive a social similarity metric from TwHIN. The network embedding fuses the heterogeneous engagements into a unified vector space that's easy to operate on.

Embedding TwHIN Nodes. We seek to learn shallow embedding vectors (i.e., vector of learnable parameters) for each user (u_j) and Tweet (t_k) in the TwHIN; we denote these learnable embeddings for users and Tweets as $\mathbf{u_j}$ and $\mathbf{t_k}$ respectively. While our approach is agnostic to the exact methodology used to embed TwHIN, we follow the approach outlined in [35, 36]. A user-Tweet pair for a particular relation type $\phi((u_j, t_k)) = r_m$ is scored with a scoring function of the form $f(u_j, t_k, r_m)$. Our training objective seeks to learn $\mathbf{u}$, $\mathbf{t}$ and $\mathbf{r}$ parameters that maximize a log-likelihood constructed from the scoring function for $(u, t) \in G$ and minimize for $(u, t) \notin G$.

For simplicity, we apply a simple dot product comparison between user and Tweet representations. For a user-tweet edge (u_j, t_k) of relation r_m, this operation is defined by:

$$f(e) = f(u_j, t_k, r_m) = (\mathbf{u_j} + \mathbf{r_m})^\intercal \mathbf{t_k} \tag{3.1}$$

As seen in Equation 3.1, we co-embed users and Tweets and scoring is performed by applying an engagement-specific translation embedding to user representations and computing the dot product with a Tweet representation. The task is then formulated as an edge (or link) prediction task. Following previous method [35, 37, 38], we maximize the following negative sampling objective:

$$\underset{\mathbf{u},\mathbf{r},\mathbf{t}}{\operatorname{argmax}} \sum_{e \in G} \left[\log \sigma(f(e)) + \sum_{e' \in N(e)} \log \sigma(-f(e')) \right] \tag{3.2}$$

where $N(e)$ is a set of negatively sampled edges by corrupting positive edges via replacing either the user or Tweet in an edge with a negatively sampled user or Tweet. As user-Tweet engagement graphs are very sparse, randomly corrupting an edge in the graph is very likely to be a 'negative' edge absent from the graph.

Equation 3.2 represents the log-likelihood of predicting a binary "real" or "fake" label for the set of edges in the network (real) along with a set of the "fake" negatively sampled edges. To maximize the objective, we learn $\mathbf{u}$ and $\mathbf{i}$ parameters to differentiate positive edges from

negative, unobserved edges.

We adopt the PyTorch-Biggraph [39] framework for scalability. Following previous approaches, we train for 10 epochs and perform negative sampling both uniformly and proportional to entity prevalence in TwHIN [39, 40]. Optimization is via Adagrad.

Upon learning dense representations of nodes in TwHIN, we utilize the Tweet representations to mine socially similar Tweets.

Mining Similar Tweet Pairs. Given the learned TwHIN Tweet embeddings, we seek to identify pairs of Tweets with *similar social appeal* – that is, Tweets that appeal to (i.e., are likely to be engaged with) similar users. We will use these socially-similar Tweet pairs as self-supervision when training TwHIN-BERT. To identify these pairs, we perform an approximate nearest neighbor (ANN) search in the TwHIN embedding space. To efficiently perform the search over 1B+ Tweets, we use the optimized FAISS[4] toolkit [41] to create a compact index of Tweets keyed by their engagement-based TwHIN embeddings. As each Tweet embedding is 256-dimensional, storing billion-scale Tweet embeddings would require more than one TB of memory. To reduce the size of the index such that it can fit on a 16 A100 GPU node, with each GPU possessing 40GB of memory, we apply product quantization [42] to discretize and reduce embeddings size. The resultant index corresponds to `OPQ64,IVF65536,PQ64` in the FAISS index factory terminology.

After creating the FAISS index and populating it with TwHIN Tweet embeddings, we search the index using Tweet embedding queries to find pairs of similar Tweets (t_i, t_j) such that $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$ are close in the embedding space as defined by their cosine distance. To ensure high recall, we query the FAISS index with 2000 probes. Finally, we select the k closet Tweets with the cosine distance between the query Tweet and retrieved Tweets' embeddings. These pairs are used in our social objective when pre-training TwHIN-BERT.

3.2.2 Pre-training Objectives

Given the mined *socially similar* Tweets, we describe our language model training process. To train TwHIN-BERT, we first run the Tweets through the language model and then train the model with a joint contrastive social loss and masked language model loss.

Tweet Encoding with LM. We use a Transformer language model to encode each Tweet. Similar to BERT [10], given the tokenized text $\boldsymbol{w}_t = [w_1, w_2, ..., w_n]$ of a Tweet t, we add special tokens to mark the start and end of the Tweet: $\hat{\boldsymbol{w}}_t = [\text{CLS}]\boldsymbol{w}_t[\text{SEP}]$. As the Tweets

[4] https://github.com/facebookresearch/faiss

are usually shorter than the maximum sequence length of a language model, we group multiple Tweets and feed them together into the language model when possible. We then apply *CLS-pooling*, which takes the [CLS] token embedding of each Tweet. These Tweet embeddings are passed through an MLP projection head for the *social loss* computation.

$$[\boldsymbol{e}_{t_1}, \boldsymbol{e}_{t_2}, ...] = \text{Pool}\left(\text{LM}([\hat{\boldsymbol{w}}_{t_1}, \hat{\boldsymbol{w}}_{t_2}, ...])\right) \tag{3.3}$$

$$\boldsymbol{z}_t = \text{MLP}(\boldsymbol{e}_t) \tag{3.4}$$

Contrastive Social Loss. We use a contrastive loss to let our model learn whether two Tweets are socially similar or not. For each batch of B socially similar Tweet pairs $\{(t_i, t_j)\}_B$, we compute the *NT-Xent* loss [43] with in-batch negatives:

$$\mathcal{L}_{\text{social}}(i, j) = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j))/\tau}{\sum_{\mathcal{N}_B(i)} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)} \tag{3.5}$$

The negatives $\mathcal{N}_B(i)$ of Tweet t_i are the $(2B-1)$ other Tweets in the batch that are not paired with t_i. We use cosine similarity for function $\text{sim}(\cdot, \cdot)$. τ is the loss temperature.

Our overall pre-training objective is a combination of the contrastive social loss and the masked language model loss [10]:

$$\mathcal{L} = \mathcal{L}_{\text{social}} + \lambda \mathcal{L}_{\text{MLM}} \tag{3.6}$$

λ is a hyperparameter that balances the social and language loss.

3.2.3 Pre-training Setup

Model Architecture. We use the same Transformer architecture as BERT [10] for our language model. We adopt the XLM-R [28] tokenizer, which offers good capacity and coverage in all languages. The model has a vocabulary size of 250K. The max sequence length is set to 128 tokens. The detailed model setup can be found in Appendix A.1.2. Note that although we have chosen this specific architecture, our social objective can be used in conjunction with a wide range of language model architectures.

Pre-training Data. We collect 7 billion Tweets in 100 languages from Jan. 2020 to Jun. 2022. Additionally, we collect 100 billion user-Tweet social engagement data covering 1 billion of our Tweets. We re-sample based on language frequency raised to the power of 0.7 to mitigate the under-representation of low-resource languages.

Training Procedure. Our training has two stages. In the first stage, we train the model from scratch using the 6 billion Tweets without user engagement. The model is trained for 500K steps on 16 Nvidia A100 GPUs (a2-megagpu-16g) with a total batch size of 6K. In the second stage, the model is trained for another 500K steps on the 1 billion Tweets with the joint MLM and social loss. We use mixed precision during training. Overall pre-training takes approximately five days for the base model and two weeks for the large model. We refer readers to Appendix A.1.2 for the detailed hyperparameter setup.

3.3 EXPERIMENTS

In this section, we discuss baseline specifications, evaluation setup, and results from two families of downstream evaluation tasks.

3.3.1 Evaluated Methods

We evaluate TwHIN-BERT against the following baselines.

- **mBERT** [10] is the multilingual language variant of the popular BERT [10] language model. It is a general domain language model trained on Wikipedia dumps.
- **XLM-R** [28] is a state-of-the-art general domain multilingual language model at its sizes. It is trained on over two terabytes of CommonCrawl data.
- **BERTweet** [30] is the previous state-of-the-art English tweet language model. It adopts a monolingual tokenizer trained from scratch on tweets and replicates RoBERTa [44] training from scratch on 845M English tweets.
- **XLM-T** [31] is a multilingual Twitter language model based on XLM-R [28]. It adopts the XLM-R tokenizer and model checkpoint and continues training on over 200M multilingual tweets.
- **TwHIN-BERT-MLM** is an ablation of our model. It is trained on the same corpus and with the same protocol as our main models. It uses only an MLM objective.

We include *base* and *large* sizes of our model train on the same corpus. All baselines are *base* variants (with between 135M to 278M parameters depending on the size of the tokenizer). Our large model has around 550M parameters.

We note that all externally published models we compared against were trained on different quantities of data and the data differed temporally and linguistically. As such, we include these models to demonstrate the gap in performance between widely-used published and open-sourced models and the model we plan on open-sourcing. On the other hand, our

Table 3.1: Engagement prediction HITS@10 on high-resource languages and overall average.

Method	Overall Average	High-Resource			
		en	**ja**	**es**	**ar**
BERTweet	-	.1414	-	-	-
mBERT	.0732	.0633	.0227	.0575	.0532
XLM-R	.0849	.0850	.0947	.0704	.0546
XLM-T	.1043	.1181	.1079	.1103	.1403
TwHIN-BERT					
- **Base-MLM**	.1161	.1400	.1413	.1204	.1640
- **Base**	.1436	.1552	.2065	.1618	**.2206**
- **Large**	**.1497**	**.1585**	**.2325**	**.2055**	.1989

Table 3.2: Engagement prediction HITS@10 on mid-resource languages.

Method	Mid-Resource			
	el	**ur**	**tl**	**nl**
BERTweet	-	-	-	-
mBERT	.0496	.0437	.0610	.0616
XLM-R	.0628	.0315	.0653	.0650
XLM-T	.0562	.0352	.0877	.0762
TwHIN-BERT				
- **Base-MLM**	.0801	.0547	.0700	.0965
- **Base**	.0944	.0627	.1030	**.1346**
- **Large**	**.1065**	**.0667**	**.1053**	.1248

base-MLM model draws a direct comparison and isolates the effect of social engagement on the resultant model.

3.3.2 Social Engagement Prediction

Our first benchmark task is *social engagement prediction*. This task aims to evaluate how well the pre-trained language models capture the social aspects of user-generated text. In our task, we predict whether users modeled via a user embedding vector will perform a certain social engagement on a given Tweet.

We use different pre-trained language models to generate representations for Tweets, and then feed these representations into a simple prediction model alongside the corresponding user representation. The engagement prediction model is trained to predict whether a user will engage with a specific Tweet. The LM-generated embeddings are fixed when we train

Table 3.3: Engagement prediction HITS@10 on low-resource languages.

	Low-Resource			
Method	**no**	**te**	**da**	**ps**
BERTweet	-	-	-	-
mBERT	.0731	.0279	.1060	.0522
XLM-R	.1661	.0505	.1150	.0727
XLM-T	.1156	.0728	.1167	.0662
TwHIN-BERT				
- **Base-MLM**	.1502	.0883	.1334	.0600
- **Base**	.1920	.1017	.1470	.0799
- **Large**	**.2118**	**.1654**	**.1475**	**.0817**

the downstream engagement prediction model.

Dataset. To curate our Tweet-Engagement dataset, we select the 50 popular languages on Twitter and sample 10,000 (or all if the total number is less than 10,000) Tweets of each language from a fixed time period. All Tweets are available via the Twitter public API. We then collect the user-Tweet engagement records associated with these Tweets. There are, on average, 29K engagement records per language. We ensure that there is no overlap between the evaluation and pre-training datasets.

Each engagement record consists of a pre-trained 256-dimensional user embedding [35] and a Tweet ID that indicates the user has engaged with the given Tweet. To ensure privacy, each user embedding appears only once, however, each tweet may be engaged by multiple users. We split the Tweets into train, development, and test sets with a 0.8/0.1/0.1 ratio, and then collect the respective engagement records for each subset.

Prediction Model. Given a pre-trained language model, we use it to generate an embedding for each Tweet t given its content $\boldsymbol{w}_t$: $\boldsymbol{e}_t = \text{Pool}\left(\text{LM}(\boldsymbol{w}_t)\right)$.

We apply the following pooling strategies to calculate the Tweet embedding from the language model. First, we take [CLS] token embedding as the first part. Then, we take the average token embedding of non-special tokens as the second part. The two parts are concatenated to form the *Combined* embedding of a Tweet.

With LM-derived Tweet embeddings, pre-trained user embeddings, and the user-Tweet engagement records, we build an engagement prediction model $\boldsymbol{\Theta} = (\boldsymbol{W}_t, \boldsymbol{W}_u)$. Given a user u and a Tweet t, the model projects the user embedding $\boldsymbol{e}_u$ and the Tweet embedding $\boldsymbol{e}_t$ into the same space, and then calculates the probability of engagement:

$$\boldsymbol{h}_u = \boldsymbol{W}_u^T \boldsymbol{e}_u, \quad \boldsymbol{h}_t = \boldsymbol{W}_t^T \boldsymbol{e}_t \tag{3.7}$$

$$P(t \mid u) = \sigma\left(\boldsymbol{h}_u^T \boldsymbol{h}_t\right) \tag{3.8}$$

We optimize a negative sampling loss on the training engagement records R. For each engagement pair $(u, t) \in R$, the loss is:

$$\log \sigma\left(\boldsymbol{h}_u^T \boldsymbol{h}_t\right) + \mathbb{E}_{t' \sim P_n(R)} \log \sigma\left(-\boldsymbol{h}_u^T \boldsymbol{h}_{t'}\right) \tag{3.9}$$

where $P_n(R)$ is a negative sampling distribution. We use the frequency of each Tweet in R to the power of 3/4 for this distribution.

Our prediction model closely resembles classical link prediction models such as [45]. We keep the model simple, making sure it will not overpower the language model embeddings.

Evaluation Setup and Metrics. We conduct a hyperparameter search on the English development dataset and use these hyperparameters for the other languages. The prediction model projects user and Tweet embedding to 128 dimensions. We set the batch size to 512, and the learning rate to 1e-3. The best model on the validation set is selected for test set evaluation.

In the test set, we pair each user with 1,000 Tweets: one Tweet they have engaged with, and the rest are randomly sampled negatives. The model ranks the Tweets by the predicted probability of engagement, and we evaluate with HITS@10. We report median results from 6 runs with different initialization.

Results. We show summarized results for selected high, mid, and low-resource languages (determined by language frequency on Twitter) in Table 3.1 3.2 3.3. Language abbreviations are ISO language codes[5]. We also show the average results from all 50 languages in the evaluation dataset and leave the details in Table 3.8. Our TwHIN-BERT model demonstrates significant improvement over the baselines on the social engagement task. Comparing our model to the ablation without the social loss, we can see the contrastive social pre-training provides a significant lift over just MLM pre-training for social engagement prediction. An analysis of all 50 evaluation languages shows the *large* model to perform better than the *base* model on average, with more wins than losses. Additionally, we also observe that our method yields the most improvement when using the *Combined* [CLS] token and average non-special token embedding. We believe the [CLS] token embedding from our model captures the social aspects of the Tweet while averaging the other token embeddings captures the semantic

[5] https://www.iso.org/iso-639-language-codes.html

Table 3.4: Text classification dataset statistics. *Statistics for Hashtag shows the numbers for each language.

Dataset	Lang.	Label	Train	Dev	Test
SE2017	en	3	45,389	2,000	11,906
SE2018-en	en	20	45,000	5,000	50,000
SE2018-es	es	19	96,142	2,726	9,969
ASAD	ar	3	137,432	15,153	16,842
COVID-JA	ja	6	147,806	16,394	16,394
SE2020-hi	hi+en	3	14,000	3,000	3,000
SE2020-es	es+en	3	10,800	1,200	3,000
Hashtag	multi	500*	16,000*	2,000*	2,000*

Table 3.5: Multilingual hashtag prediction Macro-F1 on high, mid, low resource, and average of all languages.

	High-Resource				**Mid-Resource**				**Low-Resource**				**All**
Method	**en**	**ja**	**es**	**ar**	**el**	**ur**	**tl**	**nl**	**no**	**te**	**da**	**ps**	**Avg.**
BERTweet	59.01	-	-	-	-	-	-	-	-	-	-	-	-
mBERT	54.56	68.43	42.48	38.48	44.00	36.44	52.96	39.75	46.09	49.54	59.54	29.41	50.05
XLM-R	53.90	69.07	43.80	37.85	43.94	37.56	52.99	40.85	48.94	51.47	60.35	34.92	50.86
XLM-T	55.08	70.55	45.85	42.27	44.15	39.22	54.86	41.01	49.22	52.45	59.97	33.27	51.74
TwHIN-BERT													
- Base-MLM	58.38	72.66	48.41	43.08	46.89	41.53	56.76	42.36	49.60	51.13	61.00	35.37	53.66
- Base	59.31	**73.03**	48.59	44.24	**47.59**	42.81	57.33	42.69	51.11	56.66	60.33	36.21	54.62
- Large	**60.07**	72.91	**49.88**	**45.41**	47.43	**43.39**	**59.43**	**44.80**	**51.34**	**57.03**	**61.56**	**38.24**	**55.23**

aspects of the Tweet. Naturally, utilizing both aspects is essential to better model a Tweet's appeal and a user's inclination to engage with a Tweet.

3.3.3 Tweet Classification

Our second collection of downstream tasks is Tweet classification. In these tasks, we take as input the Tweet text and predict discrete labels corresponding to the label space for each task.

Datasets. We curate a multilingual Tweet hashtag prediction dataset (available via Twitter public API) to comprehensively cover the popular languages on Twitter. In addition, we evaluate on five external benchmark datasets for tasks such as sentiment classification, emoji prediction, and topic classification in selected languages. We show the dataset statistics in Table 3.4.

• **Tweet Hashtag Prediction** dataset is a multilingual hashtag prediction dataset we

Table 3.6: External classification benchmark results.

Method	SE2017 en	SE2018 en	SE2018 es	ASAD ar	COVID-JA ja	SE2020 hi+en	SE2020 es+en	Avg.
BERTweet	72.97	33.27	-	-	-	-	-	-
mBERT	66.17	27.73	19.19	69.08	80.57	66.55	45.31	53.51
XLM-R	71.15	30.94	21.05	79.09	81.67	69.59	48.97	57.49
XLM-T	72.01	31.97	21.49	80.70	81.48	70.94	51.06	58.52
TwHIN-BERT								
- **Base-MLM**	72.10	32.44	21.79	80.48	82.12	72.42	51.67	59.00
- **Base**	72.30	32.41	22.23	80.73	82.37	71.30	54.32	59.38
- **Large**	**73.10**	**33.31**	**22.80**	**81.19**	**82.50**	**73.08**	**54.47**	**60.06**

collected from Tweets. It contains Tweets from 50 popular languages. For each language, the 500 most popular hashtags were selected, and 100k tweets with those hashtags were sampled. We ensured each Tweet will only contain one of the 500 candidate hashtags. Similar to the work proposed in [46], the task is to predict the hashtag used in the Tweet.

- **SemEval2017** task 4A [47] is a English Tweet sentiment analysis dataset. The labels are three-point sentiments of "positive", "negative", "neutral".
- **ASAD** [48] is an Arabic Tweet sentiment dataset with the same three-point labels as SemEval2017 T4A.
- **SemEval2020** task 9 [49] contains code-mixed Tweets of Hindi + English and Spanish + English. We use the three-point sentiment analysis part of the dataset for evaluation.
- **SemEval2018** task 2 [50] is an emoji prediction dataset in both English and Spanish. The objective is to predict the most likely used emoji in a Tweet.
- **COVID-JA** [51] is a Japanese Tweets classification dataset. The objective is to classify each Tweet into one of the six pre-defined topics around COVID-19.

Setup and Evaluation Metrics. We use the standard language model fine-tuning method as described in [10] and apply a linear prediction layer on top of the pooled output of the last transformer layer. Each model is fine-tuned for up to 30 epochs, and we evaluate the best model from the training epochs on the test set based on the development set performance. The fine-tuning hyperparameter setup can be found in Appendix A.1.2. We report the median results from 3 fine-tuning runs with different random seeds. Results are the evaluation metrics recommended for each benchmark dataset or challenge (Appendix A.1.3). For hashtag prediction datasets, we report macro-F1 scores. We conduct data contamination tests using character-level 50-gram overlaps [52] and found 1.56% and 1.10% contamination in the average results reported in Table 3.5 and Table 3.6.

Multilingual Hashtag Prediction. In Table 3.5, we show macro F1 scores on selected languages from our multilingual hashtag prediction dataset. We also report the average performance of all 50 languages in the dataset, and leave detailed results in Table 3.9. We can see that TwHIN-BERT significantly outperforms the baseline methods at the same *base* size. Our *large* model is slightly better than or on par with the *base* model, with a better overall performance. On the English dataset, our model outperforms the BERTweet monolingual language model trained exclusively on English Tweets and with a dedicated English tokenizer. Comparing our model to the ablation with no social loss, the two models demonstrate similar performance with our model being slightly better. These results show that while our model has a major advantage on social tasks, it retains high performance on semantic understanding applications.

External Classification Benchmarks. As shown in Table 3.6, our TwHIN-BERT matches or outperforms the multilingual baselines on the established classification benchmarks. BERTweet fares better than our *base* model with its dedicated large English tokenizer and monolingual training. Our *large* model outperforms all the baselines. We note that it is not uncommon for a monolingual PLM to perform better than its multilingual counterpart, as observed in [53, 54, 55]. Similar to hashtag prediction, TwHIN-BERT performs on par with or slightly better than the MLM-only ablation.

3.3.4 Varying Downstream Supervision

In this set of experiments, we study how TwHIN-BERT performs when the amount of downstream supervision changes. We fine-tune our model and baseline models on the hashtag prediction dataset (Section 3.3.3). We select English and Japanese as they are the most popular languages on Twitter. We change the number of training examples given to the models during fine-tuning. It is varied from 2 to 32 labeled training examples per class. We follow the same protocols as Section 3.3.3 and report macro F1 scores on the test set.

Figure 3.4 shows the results. TwHIN-BERT holds significant performance gain across different amount of downstream supervision. Note that when supervision is scarce, e.g., two labeled training examples per class given, our model has an even larger relative performance improvement over the baselines. The results indicate that our model may empower weakly supervised applications on Tweet natural language understanding.

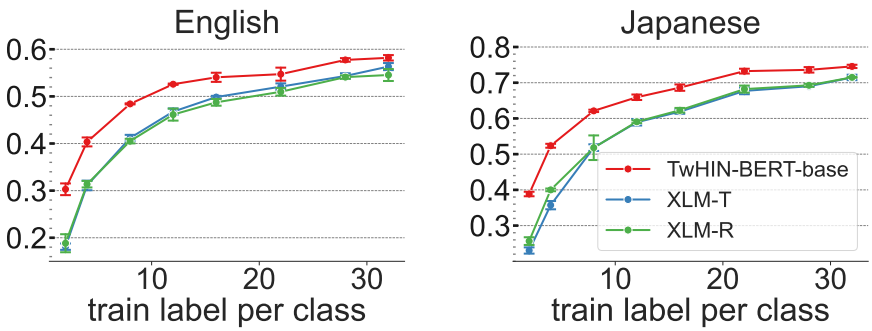


Figure 3.4: Macro-F1 on English and Japanese hashtag prediction datasets w.r.t. the number of labeled training examples per class.

Table 3.7: Feature-based classification on hashtag prediction datasets (Macro-F1).

Method	English	Japanese	Arabic
BERTweet	48.56	-	-
XLM-R	30.88	41.14	21.55
XLM-T	41.66	51.56	32.46
TwHIN-BERT-base	51.16	**64.12**	37.20
TwHIN-BERT-large	**54.12**	64.03	**38.78**

3.3.5 Feature-based Classification

In addition to language model fine-tuning experiments, we evaluate TwHIN-BERT's performance as a feature extractor. We use the hashtag prediction datasets (Section 3.3.3) and select three popular languages with different scripts. We use our model and the baseline models to embed each Tweet into a feature vector and train a Logistic Regression classifier with the fixed feature vectors as input.

Table 3.7 shows TwHIN-BERT outperforming the baselines with a wide margin on all languages. This not only shows TwHIN-BERT has learned superior Tweet representations but also showcases its potential in other feature-based downstream applications.

3.4 RELATED WORKS

Pre-trained Language Models. Since their introduction [10, 56], pre-trained language models have enjoyed tremendous success in all aspects of natural language processing. Follow-up research has advanced PLMs by further scaling them with respect to the number of parameters and training data. PLM models have grown considerably in their sizes, from millions [10, 57] of parameters to billions [58, 59, 60] and even trillion-level [61]. Another

Table 3.8: Full social engagement prediction results (HITS@10) on all evaluation Languages.

				TwHIN-BERT		
Language	**mBERT**	**XLM-R**	**XLM-T**	**Base-MLM**	**Base**	**Large**
English (en)	.0633	.0850	.1181	.1400	.1552	**.1585**
Japanese (ja)	.0227	.0947	.1079	.1413	.2065	**.2325**
Turkish (tr)	.0348	.0476	.1180	**.1268**	.1204	.0547
Spanish (es)	.0575	.0704	.1103	.1204	.1618	**.2055**
Arabic (ar)	.0532	.0546	.1403	.1640	**.2206**	.1989
Portuguese (pt)	.0731	.1285	.1709	.1201	**.1924**	.1915
Persian (fa)	.0556	.1621	.1754	.1903	.2065	**.2097**
Korean (ko)	.0275	.1105	.1446	.1675	.3611	**.3714**
French (fr)	.0488	.0635	.0805	.0700	.1030	**.1053**
Russian (ru)	.0889	.1482	.1530	.0990	**.1726**	.1704
German (de)	.0852	.1071	.3019	.2189	**.3020**	.2621
Thai (th)	.0659	.1027	.1056	.1196	**.2083**	.2004
Italian (it)	.0586	.0769	.1237	.1478	.1699	**.1706**
Hindi (hi)	.0870	.0838	.1140	.1054	.1737	**.1751**
Indonesian (id)	.0809	.0735	.0921	.1014	.1021	**.1115**
Polish (pl)	.0867	.0835	.1031	.1402	**.1696**	.1633
Urdu (ur)	.0437	.0315	.0352	.0547	.0627	**.0667**
Filipino (tl)	.0610	.0653	.0877	.1045	.1332	**.1400**
Egpt. Arabic (arz)	.0669	.0749	.1049	.0943	**.1159**	.1122
Greek (el)	.0496	.0628	.0562	.0801	.0944	**.1065**
Serbian (sr)	.1013	.1144	.1359	.1394	**.1647**	.1556
Dutch (nl)	.0616	.0650	.0762	.0965	**.1346**	.1248
Hebrew (he)	.0392	.0433	.0441	.0499	.0550	**.0577**
Ukrainian (uk)	.0497	**.0842**	.0669	.0711	.0811	**.0842**
Catalan (ca)	.1339	.1364	.1650	.1930	**.1955**	.1713
Swedish (sv)	.0942	.0716	.1161	.1342	**.1467**	.1462
Tamil (ta)	.0556	.0691	.0929	.1005	.1037	**.1060**
Finnish (fi)	.0876	.1067	.1317	.1529	.1710	**.1809**
Czech (cs)	.1155	.0904	.0766	.0997	.1062	**.1308**
Nepali (ne)	.0421	.0555	.0486	.0589	.0787	**.0851**
Azerbaijani (az)	.1561	.1148	.1702	.1576	.1712	**.1839**
Marathi (mr)	.0506	.0600	.0519	.0597	.0780	**.0906**
Bangla (bn)	.1361	.1350	.1320	.1601	.1649	**.1675**
Norwegian (no)	.0731	.1661	.1156	.1502	.1920	**.2118**
Telugu (te)	.0279	.0505	.0728	.0883	.1017	**.1654**
Pashto (ps)	.0522	.0727	.0662	.0600	.0799	**.0817**
Danish (da)	.1060	.1150	.1167	.1334	.1470	**.1475**
Vietnamese (vi)	.0929	.1060	.1085	.1216	.1417	**.1809**
Cen. Kurdish (ckb)	.0725	.0699	.0946	.1023	.1023	**.1185**
Gujarati (gu)	.0666	.0676	.0676	.0793	.1054	**.1057**
Macedonian (mk)	.0685	.0945	.0534	.0973	**.1089**	.1041
Cebuano (ceb)	.1222	.1267	.1767	.1900	.2003	**.2334**
Romanian (ro)	.1718	.1493	.1991	.2071	**.2264**	**.2264**
Kannada (kn)	.0552	.1355	.0814	.1098	.1282	**.2113**
Latvian (lv)	.0480	.0297	.0493	.0642	.0655	**.0750**
Bulgarian (bg)	.1953	.0448	.0702	.1790	.2248	**.2269**
Sinhala (si)	.0504	.0142	.0378	.0630	**.0709**	.0661
Icelandic (is)	.0319	.0341	.0466	.0364	.0387	**.0603**
Sindhi (sd)	.0619	.0288	.0553	.0885	.0951	**.0973**
Amharic (am)	.0293	.0663	.0491	.0543	.0698	**.0818**
Average	.0732	.0849	.1043	.1161	.1436	**.1497**

Table 3.9: Full hashtag prediction results (Macro-F1) on all evaluation languages.

				TwHIN-BERT		
Language	**mBERT**	**XLM-R**	**XLM-T**	**Base-MLM**	**Base**	**Large**
English (en)	54.56	53.90	55.08	58.38	59.31	**60.07**
Japanese (ja)	68.43	69.07	70.55	72.66	**73.03**	72.91
Turkish (tr)	42.87	46.37	47.14	48.72	49.31	**51.12**
Spanish (es)	42.48	43.80	45.85	48.41	48.59	**49.88**
Arabic (ar)	38.48	37.85	42.27	43.08	44.24	**45.41**
Portuguese (pt)	47.81	50.33	51.98	52.15	52.98	**56.08**
Persian (fa)	43.39	45.04	45.25	46.02	47.46	**47.94**
Korean (ko)	75.46	77.73	78.45	79.49	79.11	**80.02**
French (fr)	40.37	40.81	41.89	44.43	45.40	**47.01**
German (de)	40.80	41.42	41.11	41.32	41.38	**42.59**
Thai (th)	44.10	56.27	57.40	58.25	58.80	**59.46**
Italian (it)	42.36	41.82	42.76	45.11	44.18	**45.72**
Hindi (hi)	49.84	51.92	52.58	55.17	55.28	**57.29**
Chinese (zh)	72.88	72.54	72.40	73.85	**73.94**	72.30
Polish (pl)	48.97	50.20	50.50	51.20	51.81	**54.49**
Urdu (ur)	36.44	37.56	39.22	41.53	42.81	**43.39**
Filipino (tl)	52.96	52.99	54.86	56.76	57.33	**59.43**
Greek (el)	44.00	43.94	44.15	46.89	**47.59**	47.43
Serbian (sr)	42.50	42.32	40.71	44.22	45.95	**47.45**
Dutch (nl)	39.75	40.85	41.01	42.36	42.69	**44.80**
Catalan (ca)	48.61	47.85	48.79	51.72	52.60	**52.90**
Swedish (sv)	47.79	47.80	47.31	49.39	51.28	**51.44**
Tamil (ta)	48.04	49.67	50.65	52.85	54.14	**54.92**
Finnish (fi)	45.28	45.28	44.03	43.98	45.59	**46.42**
Czech (cs)	53.03	52.60	52.89	55.01	55.93	**56.02**
Nepali (ne)	44.58	47.00	46.94	49.83	**51.57**	51.12
Marathi (mr)	50.85	48.40	51.44	54.18	**55.76**	55.31
Malayalam (ml)	38.43	42.20	42.77	44.72	**45.86**	44.36
Bangla (bn)	57.79	57.08	56.74	59.11	60.32	**60.92**
Hungarian (hu)	60.29	59.94	60.08	61.86	**63.81**	62.68
Slovenian (sl)	58.79	59.68	59.13	61.18	62.34	**62.74**
Norwegian (no)	46.09	48.94	49.22	49.60	51.11	**51.34**
Telugu (te)	49.54	51.47	52.45	55.13	56.66	**57.03**
Pashto (ps)	29.41	34.92	33.27	35.37	36.21	**38.24**
Danish (da)	59.54	60.35	59.97	61.00	60.33	**61.56**
Cen. Kurdish (ckb)	40.28	37.59	39.06	42.89	**45.65**	45.26
Gujarati (gu)	52.55	54.09	54.24	57.36	**58.59**	58.54
Romanian (ro)	71.24	71.53	72.34	73.17	73.25	**73.58**
Kannada (kn)	54.19	55.76	56.68	59.09	**61.34**	60.19
Estonian (et)	57.81	58.10	59.00	61.95	61.22	**62.61**
Latvian (lv)	58.03	55.18	57.53	58.43	59.52	**61.47**
Bulgarian (bg)	65.20	65.52	66.45	67.42	66.94	**68.35**
Sinhala (si)	37.71	40.17	42.77	45.72	**47.54**	47.06
Icelandic (is)	51.32	48.53	50.16	53.39	**55.53**	54.61
Sindhi (sd)	27.28	26.46	31.08	32.42	35.05	**35.28**
Basque (eu)	58.55	56.55	56.78	59.56	60.62	**61.04**
Amharic (am)	24.10	28.57	35.01	34.20	**37.47**	36.87
Lithuanian (lt)	71.31	69.50	69.65	72.26	72.43	**73.09**
Welsh (cy)	58.36	58.50	57.56	59.66	59.95	**60.24**
Haitian Creole (ht)	68.13	67.05	67.97	70.70	71.33	**71.41**
Average	50.05	50.86	51.74	53.66	54.62	**55.23**

avenue of improvement has been improving the training objectives used to train PLMs. A broad spectrum of pre-training objectives have been explored with different levels of success, Notable examples include masked language modeling [10], auto-regressive causal language modeling [57], model-based denoising [62], and corrective language modeling [63]. Despite these innovations in scaling and pre-training objectives, the vast majority of the work has focused on text-only training objectives applied to general domain corpora, e.g., Wikipedia and CommonCrawl. In this paper, we deviate from most previous works by exploring PLM training using solely Twitter in-domain data and training our model based on both text-based and social-based objectives.

Tweet Language Models. While a majority of PLMs are trained on general domain corpora, a few language models have been proposed specifically for Twitter and other social media platforms. BERTweet [30] mirrors BERT training on 850 million English Tweets. TimeLMs [64] trains a set of RoBERTa [44] models for English Tweets on different time ranges. XLM-T [31] continues the pre-training process from an XLM-R [28] checkpoint on 198 million multilingual Tweets. These methods mostly replicate existing general domain PLM methods and simply substitute the training data with Tweets. However, our approach utilizes additional social engagement signals to enhance the pre-trained Tweet representations.

Enriching PLMs with Additional Information. Several existing works use additional information for language model pre-training. ERNIE [65] and K-BERT [66] inject entities and their relations from knowledge graphs to augment the pre-training corpus. OAG-BERT [67] appends metadata of a document to its raw text, and designs objectives to jointly predict text and metadata. These works focus on bringing metadata and knowledge by injecting training instances, while our work leverages the rich social engagements embedded in the social media platform for text relevance. Recent work [68] has utilized document hyperlinks for LM pre-training, but does so with a simple three-way classification objective.

Network Embedding. Network embedding has emerged as a valuable tool for transferring information from relational data to other tasks [69]. Early network embedding methods such as DeepWalk [70] and node2vec [71] embed homogeneous graphs by performing random walks and applying SkipGram modeling. With the introduction of heterogeneous information networks [72] as a formalism to model rich multi-typed, multi-relational networks, many heterogeneous network embedding approaches were developed [73, 74, 75, 76, 77]. However, many of these techniques are difficult to scale to very large networks. In this work, we apply knowledge graph embeddings [40, 78, 79], which have been shown to be both highly scalable and flexible enough to model multiple node and edge types.

3.5 CONCLUSIONS

In this work we introduce TwHIN-BERT, a multilingual language model trained on a large Tweet corpus. Unlike previous BERT-style language models, TwHIN-BERT is trained using two objectives: (1) a standard MLM pre-training objective and (2) a contrasting social objective. We perform a variety of downstream tasks using TwHIN-BERT on Tweet data. Our experiments demonstrate that TwHIN-BERT outperforms previously released language models on both semantic and social engagement prediction tasks. We release TwHIN-BERT[6][7] to the academic community to further research in social media NLP.

[6] http://huggingface.co/Twitter/twhin-bert-base
[7] http://huggingface.co/Twitter/twhin-bert-large

Chapter 4: TEXT-RICH NETWORK MINING: WEAKLY-SUPERVISED TEXT CLASSIFICATION

Building upon the foundation of text-rich network construction and consolidation discussed in the previous chapters, we now delve deeper into the mining aspects of these networks. In the previous chapter, we explored how text-rich networks can be leveraged for language model pre-training, enabling us to capture more meaningful representations of text data. This chapter focuses on another crucial application of text-rich network mining: weakly supervised text classification.

Text classification is a fundamental task in organizing and understanding semi-structured text corpora, but traditional approaches often rely heavily on extensive human annotation. In this chapter, we leverage the inherent structure of text-rich networks to develop LTRN (Learning on Text-Rich Networks), a novel approach for minimally supervised text categorization that reduces the reliance on large amounts of labeled data.

LTRN combines a text analysis module for raw text embedding and classification with a network learning module for semantic-aware propagation of categorical information on the network. By employing co-training and feature sharing, we enable these modules to mutually enhance each other, resulting in improved text classification performance.

This work on text-rich network mining for weakly supervised text classification complements the language model pre-training approach discussed in the previous chapter, demonstrating the versatility and potential of text-rich network mining in enhancing the efficiency and effectiveness of knowledge discovery in semi-structured text corpora.

In the following sections, we will delve into the details of LTRN and demonstrate its effectiveness through extensive experiments, further solidifying the significance of text-rich network mining within the broader context of this thesis.

4.1 OVERVIEW

Text categorization is a fundamental task in organizing and understanding content gathered from the Web. The Web's ever-evolving nature constantly brings in emerging categories to real-world text categorization, leading to hundreds of or even more categories. For example, e-commerce product categorization has tens of thousands of categories [80, 81, 82]; classifying events from social media have hundreds of event types in commonly used event databases [83]. Therefore, instead of the laborious supervised setting, we set our focus on the *minimally-supervised* setting—the user only provides a handful of examples per category (in our experiments, no more than 3 per category). Our goal is to leverage these seed examples as

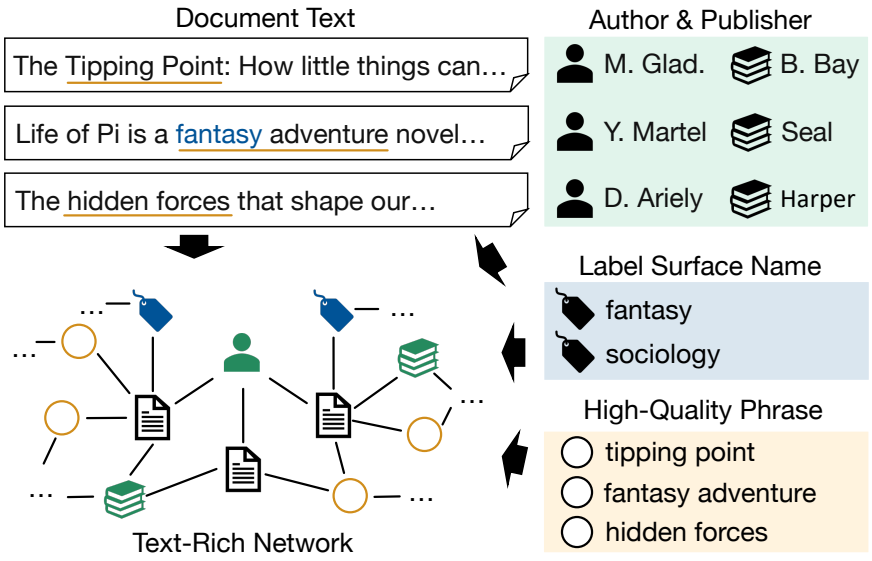


Figure 4.1: An illustrative example of a text-rich network constructed from e-books. It integrates the book descriptions (i.e., raw texts) and structure-rich metadata (e.g., author & publisher, high-quality phrases, and label surface names) together and provides a holistic view.

well as unlabeled examples that are ubiquitously available to build an accurate categorization model.

In this work, we provide a *text-rich network* prospective for minimally-supervised categorization of *structure-rich* text, i.e., text accompanied with metadata. We argue that interconnecting documents and metadata together is beneficial in a minimally supervised setting. Consider an example of book classification. Given only a book intro, such as the one to the upper left in Figure 4.1, it may be hard to tell the book's genre. While if we look at its author and publisher and find more similar "sociology" books, we may have a better chance to put it into the correct category. As illustrated in Figure 4.1, we organize a structure-rich corpus into a text-rich network, integrating textual documents and various types of metadata into a single, unified framework. It explicitly models their relations, offering a complementary view to raw text in the corpus, and enables network-based analysis.

We highlight two major challenges in modeling text-rich networks: (1) combining text and network for minimal supervision and (2) handling "mixed signal" from the text-rich network structure. First, combining text and network structure is non-trivial, especially in the context of minimal supervision. A desirable framework should take advantage of both modern natural language processing models for raw text understanding and graph learning models to propagate information on the network effectively. However, trivially gluing two deep models from both data sources together may result in an over-complicated model that

is infeasible computationally and prone to over-fitting on a small number of human labels. Second, the network is by no means entirely constructed in a *class-discriminative* way. On the one hand, the network exhibits weak homophily, i.e., nodes sharing links or similar neighbors do not necessarily belong to the same category. The problem becomes more evident as the number of categories becomes larger, and the category semantics become more similar. As an example, when categorizing e-commerce products in different categories, "screw-driver" and "wrench" products can easily be confused because they usually share similar brands, manufacturers, and have similar wording in their product descriptions, making them close in the text-rich network. On the other hand, the linkage in the network is far from perfect. Continuing the product categorization example, products will be connected to nodes that are not label-indicative, such as the connection from a product to a phrasal node "high-quality". The text-rich network's nature calls for a model that can identify label-discriminative features from the network.

While most existing methods [84, 85, 86, 87] on minimally supervised text categorization employ text data only, pioneer studies on text-rich network either assume a relatively clean network structure for learning [88, 89], or rely on additional human effort, such as user-given network motif patterns, to filter out uninteresting nodes [90, 91]. As a result, they do not address the aforementioned challenges in text-rich network modeling.

To this end, we propose LTRN, a novel framework for minimally supervised text categorization by **L**earning on **T**ext-**R**ich **N**etworks. As shown in Figure 4.2, LTRN has two data-driven modules *in parallel* – a text analysis module for raw text embedding and classification, and a network learning module for semantic-aware propagation of categorical information on the network. The two modules, one powered by BERT [10] and the other powered by a graph neural network (GNN), are designed to have very different *inductive biases*. The text module models raw text but is unaware of the network structure, while the network module effectively aggregates information from the network but relies on vectorized node features. This offers us an opportunity to let each module learn from the other. We leverage *co-training* and *feature sharing* to train both modules jointly. In each iteration of our framework, we let both modules assign pseudo labels to the unlabeled documents, and employ co-training to pool two diverse sets of high confident predictions together, forming a pseudo labeled training set for the next algorithm iteration. In this fashion, both modules can mutually enhance each other through the other model's predictions. As the GNN requires fixed, vectorized features as input, at the end of each iteration, we share the document embedding from BERT to the GNN model, ensuring the GNN model gets the up-to-date document features. After the framework is fully trained, we use the BERT model as our final categorization model. The BERT model does not hinge on the network structure, thus is more flexible in both

transductive and inductive setting when categorizing incoming new documents.

The specific design of each module in our framework is tailored to its data source. The BERT model [10] in the text module leverages sentence-level semantics for text-based classification directly, and also provides meaningful document embedding for the network module. The GNN model in the network module adopts a novel architecture with two proposed mechanisms to capture class-discriminative signals in the text-rich network. First, a personalized PageRank (PPR) based neighborhood sampling method picks the most relevant neighbors for each document node. Then an attention-based aggregation method rolls out unhelpful neighbors and further narrows down label-indicative neighbors. Moreover, the model is scalable, making it suitable for gigantic real-world datasets. Our network module design differs from the commonly used neighborhood sampling GNNs [92, 93] that do not model label-discriminativeness, and sets us apart from the popular Graph Attention Network [94] which applies attention to the full neighborhood.

In summary, our main contributions are as follows:

- We propose a joint learning framework on the text-rich network for minimally-supervised text categorization. The framework has two data-driven modules with different inductive biases, a text analysis module, and a network learning module. We leverage co-training and feature sharing to train both modules jointly.
- We propose a novel GNN architecture for text-rich network modeling. The proposed model adopts personalized PageRank-based neighborhood sampling and attentive aggregation. The model successfully handles noise on the text-rich network and scales up to large datasets.
- We conduct experiments on two real-world datasets and achieve significant gain over various competitive baselines. On the challenging product categorization dataset with ∼700 categories, our model obtains an F-measure of over 90% with only 3 seed per category. Comparing with a supervised BERT model, our model saves training labels by 20× with only $< 2\%$ sacrifice on F-measure, and it out-performs state-of-the-art self-training and augmentation-based methods trained on the same seed labels by $> 13\%$.

Reproducibility: The source code to reproduce the experiments can be found at https://github.com/xinyangz/ltrn.

4.2 PRELIMINARIES

In this section, we first formally define the problem, and then introduce the concept of the text-rich network and its construction.

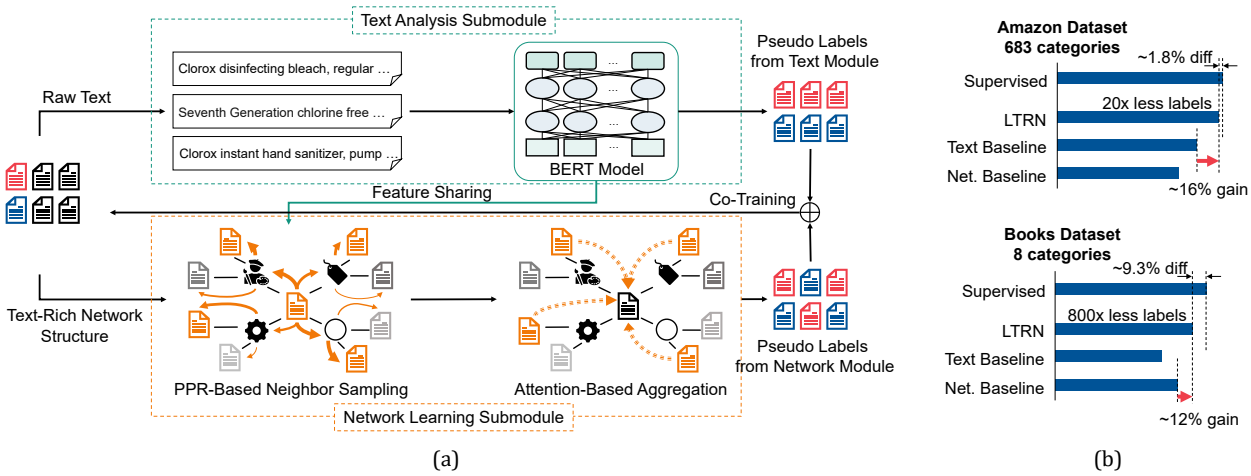


Figure 4.2: (a) An overview of LTRN. It jointly trains a text analysis module for raw text embedding and classification, and a scalable, noise-resilient network learning module for prediction on network structure. Both modules generate pseudo labels to extend minimal supervision, and are co-trained with feature sharing mechanism. (b) On two real-world datasets, with only 3 labeled seed examples per category, LTRN achieves significant gains over baselines and rivals the supervised method.

4.2.1 Problem Formulation

Structure-rich text categorization takes a set of documents $\mathcal{D} = \{d_1, ..., d_N\}$ accompanied with metadata $\mathcal{M} = \{m_1, ..., m_N\}$, a set of labels $\mathcal{L} = \{l_1, ..., l_{|\mathcal{L}|}\}$, and aims to assigns a label l_i for each document $d_i \in \mathcal{D}$. Metadata refers to the complementary attributes coming with textual documents in the data collection, such as writers and publishers of books, brands and manufacturers of e-commerce products, high-quality phrases mined from the text, and label surface names that are mentioned in the corpus. In the minimally-supervised setting, the user gives a few (in our framework, no more than 3) labeled *seed documents* per class $\mathcal{D}_L = \bigcup_{l \in \mathcal{L}} \mathcal{D}_l \subset \mathcal{D}$. Distinct from the conventional supervised setting, the number of seed documents is small $|\mathcal{D}_L| \ll |\mathcal{D}|$; both labeled $\mathcal{D}_L$ and unlabeled documents $\mathcal{D} - \mathcal{D}_L$ are adopted to learn the categorization model.

4.2.2 Text-Rich Network

To facilitate minimally-supervsed text categorization, we build a *text-rich network* from the corpus. It provides a holistic view of the raw text documents and various document metadata.

Formally, a text-rich network $G = (D, V, E, \phi, \psi, \omega)$ has a set of document raw text D, a set of nodes V, and a set of edges E. Nodes $V = (V_T, V_A)$ are divided into textual nodes V_T and

auxiliary nodes V_A, where textual nodes have associated textual descriptions and auxiliary nodes serve as bridges connecting textual nodes together. ϕ and ψ are type mappings that maps each node v to its type $\phi(v)$ and each edge e to its relation $\psi(e)$. Mapping $\omega : V_T \mapsto D$ is a one-to-one content mapping that maps each textual node to its raw text content.

Take the e-books dataset in our experiments as an example (Figure 4.1). The raw text documents are book titles and descriptions. The textual nodes in the network are the books. The auxiliary nodes are authors, publishers, high-quality phrases, and label surface names.

Specifically, the metadata nodes used in our experiments can be grouped into the following categories.

Document Attributes as Nodes. We cast all distinct document attribute values into nodes in the text-rich network. For instance, all authors and publishers in an e-book collection; all brands and manufacturers in an e-commerce product collection. The edge weights from each document to its attribute nodes are set to 1.

High-Quality Phrases as Nodes. We use AutoPhrase [95] to mine high-quality phrases from the textual documents and present them as nodes in the network. The edge weight from a phrase to a document is the TF-IDF score of that phrase in the document. People have used other strategies for text-rich network construction, such as using all words in the corpus vocabulary as nodes [91, 96]. We include only high-quality phrases into our network because they will not introduce an overwhelmingly large number of nodes and work well in practice.

Label Surface Name as Nodes. Label surface name is another important source of information when supervision is scarce. For example, on e-commerce platforms, merchants often put a product's category into its title; on movie review sites, the movie introduction may mention a movie's genre. While the occurrence of a label surface name in a document does not necessarily imply the document belongs to that category, it represents a relation between the document and the label in many cases. As opposed to distant supervised methods, we do not leverage label name matching for hard or soft labeling; instead, we add an edge between a document and a matched label surface name and set the edge weight to the label number of occurrences in the document text.

4.3 OUR FRAMEWORK

In this section, we describe our proposed framework and introduce its key components in detail.

4.3.1 Overview

The proposed LTRN is a pseudo labeling framework where the model assigns pseudo labels to unlabeled examples in each of its iterations, and gradually improves the performance through iterations. The general workflow of LTRN is shown in Figure. 4.2. After constructing a text-rich network from the corpus, LTRN relies on two major components to conduct categorization and pseudo labeling. The first component is a text analysis module. We employ the BERT model [10] in this module and use it for both document classification and embedding. The second component is a network learning module. We propose a novel GNN model for class-discriminative, scalable learning. The model takes network structure as well as node features as input, and generates prediction on textual nodes (corresponding to raw text documents) in the network. Distinct from most existing works, our framework models both raw text and network structure, and we learn both modules in parallel to let them mutually enhance each other.

Putting two modules together, we introduce a joint training method with two techniques – co-training and feature sharing. In each iteration of the framework, the text module and the network module are trained separately on both user given seed documents and pseudo labeled documents. Once both modules are updated, we generate two sets of pseudo labels from the two modules, and pool them together to update the shared pseudo label set. Thereafter, each module could benefit from its own confident predictions as well as the confident predictions from the other module by co-training on the shared pseudo label set. At the end of each iteration, we generate up-to-date document embedding using the BERT model, and share the embedding with the GNN model as features of the textual nodes. When the model is fully trained, we use the BERT model for new document categorization.

4.3.2 Network Learning Module

The network learning module aims to build a class-discriminative, scalable machine learning model for effective propagation of categorical information on the text-rich network. We propose a novel GNN architecture for the *semantic-aware* propagation of minimal supervision. The model takes both the network structure and the node features as input, and can be learned end-to-end. Once trained, the network module assigns pseudo labels to the unlabeled nodes in the network, and the most confident pseudo labels will be selected to improve the text analysis counterpart of the framework.

We highlight two design requirements of the GNN architecture: class-discriminativeness and scalability. For class-discriminativeness, the model must be able to handle the text-

rich network constructed from a non-label discriminative way. The network exhibits weak homophily, meaning that an edge in the network does not necessarily imply the two nodes connected by it belongs to the same category. Popular GNN architectures such as Graph Convolutional Networks [97] do not parameterize the feature aggregation process, making them suboptimal in networks with weak homophily, thus do not satisfy our needs. Our model must be able to identify the more "important" neighbors in a node's mixed local neighborhood, i.e., which of the neighbors are more label-indicative in terms of offering meaningful semantics for category prediction on the current node. For scalability, real-word corpora, especially unlabeled corpora collected automatically, are usually gigantic. We cannot afford full batch GNN models, as the network wouldn't fit into (GPU) memory.

To meet the aforementioned requirements, we propose a novel GNN architecture that is capable of learning class-discriminativeness on the text-rich network, in a mini-batch fashion. The model uses personalized PageRank [98] (PPR) for neighborhood sampling, and an attention mechanism for feature aggregation.

The overall architecture of the GNN model involves two stages, feature transformation and neighborhood aggregation. For each node, the model applies the following operations:

$$\boldsymbol{h}_i = f_{\text{trans}}(\boldsymbol{x}_i), \quad \boldsymbol{z}_i = g_{\text{agg}}(\boldsymbol{h}_i, \boldsymbol{H}, G) \tag{4.1}$$

where f_{trans} is a transformation function on node features, and g_{agg} is an aggregation function which aggregates hidden representation from the node's neighborhood and combines it with the target node's representation. In our model, f_{trans} is a multi-layer feed-forward neural network, and we will introduce g_{agg} in the following subsections.

PPR-based Neighborhood Sampling

The first step of feature aggregation in a GNN is to define a node's neighborhood and select proper neighbors for aggregation. Most GNN models [92, 94, 97] aggregates from a node's first-hop neighbors, and extends to multi-hop neighbors through multiple aggregation layers. As the size of the neighborhood grows exponentially with the number of hops, to scale up to large datasets, we sample a fixed number of nodes from the neighborhood.

Recent works have shown that personalized PageRank (PPR) is very effective in both graph neural networks [99, 100] and in propagating weak supervision [91]. We, therefore, adopt PPR for neighborhood sampling in GNN.

Personalized PageRank [98] can be derived from a random walk with restart process on

the network. It takes the network structure as input, and computes a ranking score $P_{i,j}$ from each node j on the network to a target node i. The larger the $P_{i,j}$, the more "similar" node j is to node i. Let $\boldsymbol{P} \in \mathbb{R}^{N \times N}$ be the personalized PageRank matrix of the graph, where each row of the matrix $\boldsymbol{P}_{i,:}$ corresponds to a PPR vector to a target node i. $\boldsymbol{P}$ is defined as the solution to the following equation:

$$\boldsymbol{P} = \beta \hat{\boldsymbol{A}} \boldsymbol{P} + (1 - \beta) \boldsymbol{I} \tag{4.2}$$

where β is the reset probability for PPR and $\hat{\boldsymbol{A}}$ is the normalized adjacency matrix. PPR is well studied and can be approximated efficiently, even for very large networks [101, 102]. Similar to [100], we use a push iteration method [102] to efficiently compute PPR scores.

After solving the PPR matrix $\boldsymbol{P}$, we define the PPR sampled neighborhood of node i as its top-K PPR neighbors:

$$\mathcal{N}(i) = \underset{V' \subset V_T, |V'| = K}{\operatorname{argmax}} \sum_{v_j \in V'} P_{i,j} \tag{4.3}$$

Note that we only select textual nodes as top PPR neighbors because only textual nodes have meaningful text embedding features. K is a small value compared to a nodes' full neighborhood size. We find $K = 50$ is good enough in our experiments.

Attention-Based Aggregation

Top neighbors sampled by PPR are usually quite high quality; however, two key issues are still unresolved. First, the PPR computation only leverages the network structure, thus is unaware of node features and semantics; second, the sampled neighbors are NOT guaranteed to be label-indicative. Instead of computing a weighted sum of neighborhood representation using edge weights or PPR scores, we would like to parameterize the aggregation process, so that the model can learn to focus on more label-indicative nodes.

We adopt an attention-based aggregation strategy:

$$\boldsymbol{z}_i = g_{\text{agg}}(\boldsymbol{h}_i, \boldsymbol{H}, G) = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{i,j} P_{i,j} \boldsymbol{h}_j\right) \tag{4.4}$$

$$\alpha_{i,j} = \text{Sigmoid}\left(\boldsymbol{W}_q \boldsymbol{h}_i \cdot \boldsymbol{W}_k \boldsymbol{h}_j\right) \tag{4.5}$$

where $\boldsymbol{W}_q$ and $\boldsymbol{W}_k$ are learnable weights. The idea is to first map hidden representation

of the target node $\boldsymbol{h}_i$ and the neighbor node $\boldsymbol{h}_j$ into a shared attention space, then compute their dot product and map it to $\alpha \in [0, 1]$. The smaller the dot product is, the less relevant the neighbor is to the categorization of the current node. The attention weight α is then used to adjust the personalized PageRank score of the neighbor $P_{i,j}$, before a weighted sum aggregation of all neighboring node representations. Our attention formulation is slightly different from that of GAT [94], as GAT does not explicitly model the dot product similarity of a pair of nodes.

4.3.3 Text Analysis Module

Our framework is compatible with any text classifier that takes raw text as input. We adopt a pre-trained BERT [10] model in this work as it has been proven to be generalizable, and we have found it to work well given a small number of labeled examples.

The BERT model is a Transformer [29] language model pre-trained on large, general domain corpora. Input sentences to the BERT model are preceded with a [CLS] token and succeeded with a [SEP] token. The model is trained using a masked language model (MLM) objective and a next sentence prediction objective. We fine-tune the BERT model with the language model objectives on the domain corpus before using it for document categorization and embedding.

Document Categorization. We append one linear layer to the model, which takes the [CLS] token embedding from the final layer of the Transformers, and produce categorical predictions on the label space. Together with the added linear layer, the whole model is fine-tuned on labeled data, with a larger learning rate for the final layer and a smaller learning rate for the Transformer layers.

Document Embedding. We adopt two strategies for document embedding generation. The first strategy works before the model is fine-tuned on labeled data, and we use this method to generate initial embedding for textual nodes in the text-rich network. We take the token embedding from the final Transformer layer, discard [CLS] and [SEP] token embedding, and take the average of the rest of the token embeddings as the sentence embedding. We discard [CLS] and [SEP] token embedding because [CLS] is used for next sentence prediction in the pre-training, making it irrelevant to categorization, and both tokens have no specific meaning. We find this strategy to work better than taking [CLS] embedding or using an average of all token embeddings as sentence representation.

The second strategy works when the model is fine-tuned on labeled data. This method is adopted from the second to the last iteration of the framework. We use the [CLS] embedding

from the final Transformer layer as the sentence embedding. This is because once the model is fine-tuned, [CLS] captures the categorical semantic of the sentence.

4.3.4 Joint Training of Text and Network Modules

Now we have introduced our network and text learning modules, we are ready to put them together through a joint training framework. Algorithm 4.1 describes the joint training procedure. We employ two techniques for training with weak supervision: co-training and feature sharing.

Co-Training. Since both of the modules in our framework are learning models, we leverage co-training to let them mutually enhance each other. The basic idea is to take confident predictions from one model and add them to the other model's training set. As two modules have different inductive biases, they usually generate different predictions, making the pseudo labeled training set diverse.

In each of the training iteration, we run both models on the unlabeled document set, and take the most confident predictions from each model. We set a confidence threshold to filter out low confident predictions, and take top M predicted documents from each category as pseudo labeled documents for that category. The choice of these parameters is described in Section 4.4.3. We pull those predictions together to form a pseudo labeled training set, and join it with the given seed documents. Both models will be trained on the new training set in the next iteration. This process is repeated until the pseudo label set stays the same, or until a maximum iteration is reached. We found the model to work well within 5 iterations.

Feature Sharing. The performance of our GNN module relies on the quality of node features. It cannot process raw text, and has to take vectorized features as input. In our framework, we generate document node features from the text module. Initially, the features are generated by an unsupervised BERT model. As the training progress, the text module gets improved, and could produce higher quality embedding for the documents. Therefore, we share the most up-to-date features from the text module with the GNN module. We have found that such a feature sharing mechanism is beneficial for the GNN model's performance.

4.4 EXPERIMENTS

In this section we answer the following research questions with experiments on real-world datasets.

- How does our model perform against state-of-the-art methods with minimal supervision?

Algorithm 4.1: Joint Training of LTRN

Input: Text-rich network G (including all Documents D), seed (labeled) documents D_S.

Initialize training document set $T = D_S$;
Initialize text module f and network module g;
Initialize document embedding $X \leftarrow \text{embed}(f, D)$;
repeat
  $f \leftarrow \text{train\_text\_module}(T)$;
  $g \leftarrow \text{train\_network\_module}(T, X, G)$;
  $T_1 \leftarrow \text{take\_confident\_prediction}(f, D - D_S)$;
  $T_2 \leftarrow \text{take\_confident\_prediction}(g, D - D_S)$;
  $X \leftarrow \text{embed}(f, D)$;
  $T \leftarrow S \cup T_1 \cup T_2$;
until *max_iter or T doesn't change*;

Table 4.1: Dataset Statistics.

	#doc	#category	#train-labeled	#train-total	#dev	#test
Amazon	104,787	683	2,049	41,914	10,479	52,394
Books	33,594	8	24	21,163	2,354	10,079

- How do the text and network modules of our framework mutually enhance each other through joint training?
- Is the GNN architecture design of our network module helpful?
- How does the number of labeled seed documents impact the model performance?

4.4.1 Datasets

We conduct our experiments on two real-world data collections: Amazon products and Goodread Books. The Books dataset is adapted from [91, 103], while the Amazon dataset is a larger scale dataset with many more categories collected by us. The dataset statistics are shown in Table 4.1. For our main experiments, we provide 3 seed documents per category as supervision. The seed documents are randomly chosen from the entire labeled training set. We will study the effects of seed document set in Section 4.4.6.

- **Amazon.** We collected ∼100K products from Amazon.com spanning 683 product categories. The document attributes include the product brand and product manufacturer (e.g., "Seventh Generation" and "Unilever Group" as the brand and manufacturer of a bleach product). The textual description of each product has three parts: (1) a title, which

is a concise summary of the product; (2) a bullet point list, highlighting product features; (3) a long description, introducing the product in more detail. We concatenate all three parts together in the above order because it follows the same ordering a typical customer views a product. If the full text is too long for the model, this concatenation strategy ensures that the model sees the most important part of the product text.

We labeled the products through a crowdsourcing platform. In this way, we ensure that the labels are high-quality. We do not rely on category information in the Amazon catalog, as that information might be machine generated and may be incorrect.

This dataset has far more categories than the datasets from prior works [86, 87, 91, 104]. Many of the categories are close to each other, e.g. "screw driver" vs. "wrench", "paper towel holder" vs. "hanging rod". The dataset will help us benchmark different methods under a fine-grained setting.

- **Books [91, 103]**. The Books dataset contains book descriptions from a popular online book review website named Goodreads[8]. We follow [91] and select a subset of books from 8 popular genres[9] from the original dataset [103], resulting in a dataset with 33,594 books. The task is to predict the genre of each book. The document attributes include the author and publisher of each book. The dataset has 22,145 distinct authors and 5,186 publishers. The textual part of each book contains its title and description. Compared to the Amazon dataset, we have a much smaller number of categories. Keeping the number of labeled seed documents the same, the dataset has only 24 labeled documents in total.

We use an inductive setting throughout our experiments. The models have access to the user-given seed documents and the unlabeled documents in the training set, while the models at training do not see the test set documents. This setting is different from the ones in [86, 91, 105] as they use a transductive setting.

4.4.2 Compared Methods

We evaluate LTRN against a variety of baseline methods. Towards handling weak supervision, we include representative works using (1) data augmentation, (2) pseudo labeling, and (3) combined methods. By input data structure, we compare against (1) text-based methods, (2) graph-based methods, and (3) combined methods.

- **BERT-supervised.** BERT [10] is the de facto pre-trained language model for natural language understanding. We fine-tune a BERT model using all the documents from the

[8] https://www.goodreads.com/

[9] Categories in Books dataset: (1) children, (2) comics, (3) fantasy, (4) history & biography, (5) crime, mystery thriller, (6) poetry, (7) romance, (8) young adult.

training set as our reference supervised model.

- **BERT-seed.** We train a BERT model using only seed documents. This sets our text-based baseline without specific techniques to handle weak supervision.
- **BERT-self-train.** We train a BERT model using the seed documents, as well as self-training on unlabeled documents. This method resembles our text-based pseudo labeling baseline.
- **VAT.** Virtual adversarial training [104] perturbs the training examples in the feature space towards the worst direction, and train the classification model on the perturbed "pseudo examples" to improve model robustness.
- **EDA** [85] is a text data augmentation method with four simple operations: synonym replacement, random insertion, random swap, and random deletion. We apply EDA to our seed document set and train a BERT model with EDA.
- **UDA** [84] is the state-of-the-art data augmentation technique for deep neural network training. It performs back translation and TF-IDF word replacement to augment both the labeled and unlabeled data. After that, it trains a BERT classifier with a supervised cross-entropy loss and an unsupervised consistency loss.
- **WeSTClass** [86] combines data augmentation and pseudo labeling by generating pseudo documents and employing self-training. It is a pre-BERT classification model. We use its CNN variant as it performs the best over all the model variants.
- **PPRGo-seed.** PPRGo [100] is a state-of-the-art scalable GNN model with personalized PageRank based neighborhood aggregation. We train it on seed document only as our network-based baseline.
- **PPRGo-self-train.** We train PPRGo with seed documents and self-training on unlabeled documents. This method resembles our network-based pseudo labeling baseline.
- **Text-GCN** [96] constructs a corpus-level network of words and documents with word-document edges based on TF-IDF and word-word edges based on PMI. It then applies the Graph Convolutional Network [97] for document categorization.
- **Text-ING** [106] converts each document into a network, where nodes are words in the documents and edges are word occurrences based on a sliding window. After converting documents to networks, a GNN is applied for classification.
- **CANE** [88] is a textual network embedding method which applies a convolutional neural network for text representation, and maximizes edge probabilities with a combination of structure-based and text-based objectives.
- **MetaCat** [105] learns embedding for words, documents, labels, and metadata to categorize text documents with minimal supervision. It combines network embedding with a CNN text classifier, and employs data augmentation.

We denote our method as **LTRN**. We use Micro-F1 (i.e., accuracy) and Macro-F1 as our evaluation metrics.

4.4.3 Model Configuration

We append metadata as a string to the main text for all text-based baselines to ensure a fair comparison. For example, if a product from the Amazon dataset has a brand "Clorox", then we append "[BRAND] Clorox" to the product's description. For PPRGo models, we use unsupervised BERT embedding as node feature.

For all methods using the BERT model, we use BERT-base architecture with pre-trained weights from the original authors and adapted by HuggingFace Transformers library[10]. We then fine-tune it using masked language model objective on domain-specific corpus with a 10^{-5} learning rate. For the PPRGo model as well as our GNN submodule, we set the number of layers to 2, and the hidden dimension to 64. We set max sequence length to 128 for all models, and ensure that the metadata string appended to the end of the text is not truncated. During BERT model fine-tuning, we set the learning rate to 10^{-4} for Transformer layers, and 10^{-2} for the classification layer.

For the CANE model, since it operates on homogeneous textual networks, we convert our heterogeneous text-rich networks to document-document networks. Denote D_M as the document to metadata edge matrix, we obtain document to document edges and their weights by $D_M D_M^T$.

As for our model, we set $K = 50$ for top K PPR neighbor sampling in GNN. We set the confidence filtering threshold to be 0.9 for co-training, and take top 50 and 500 documents for each category in Amazon and Books dataset, respectively. We use different top prediction numbers because the number of unlabeled documents per category is larger in the Books dataset. One can also use a top percentage of confident predictions for co-training. The same setting is applied to all self-training methods. We set the maximum number of iteration in our framework to be 5.

4.4.4 Main Results

We present our main experiment results in Table 4.2. Our proposed LTRN model consistently outperforms competing baseline methods on both datasets. We also note the following key observations throughout our experiments.

[10] https://huggingface.co/transformers/pretrained_models.html

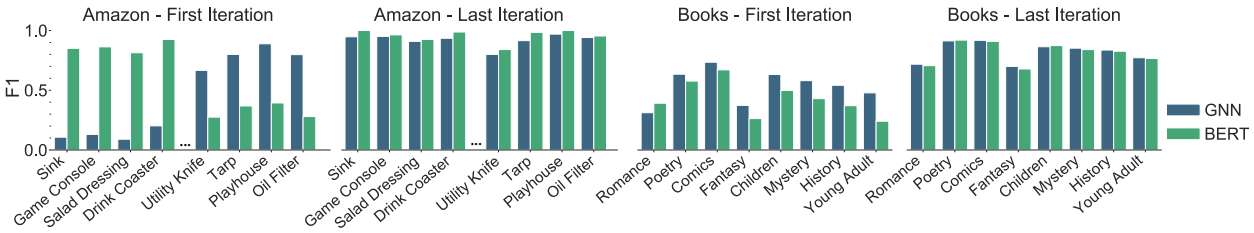


Figure 4.3: Per Category Performance Change for Text and Network Modules Over Iterations.

- On the Amazon dataset, text-based methods have an advantage over their network-based counter parts. While on the Books dataset, network-based methods achieve higher performances. The joint training framework of our model takes advantage of both sources of information, thus achieving superior performance than the baselines.
- BERT representations generalize well on both datasets even when the supervision is scarce. The models that do not use BERT representation (VAT, WeSTClass, MetaCat, Text-GCN) could suffer given very few training labels. The significant performance gain of our model over BERT baselines demonstrates that the network module enhanced BERT performance.
- Comparing Text-ING to other network-based methods, it shows inferior performance. We attribute this observation to the minimally supervised setting, which limits the training vocabulary and the Text-ING model's generalization power. CANE also suffers in our experiments, as our text-rich network has weak edge homophily. CANE relies on maximizing all edge probabilities, thus may not be ideal for this application scenario.
- Data augmentation methods have great success on the Amazon dataset, EDA being our best performing baseline. However, they struggle with the Books dataset. Comparing UDA to EDA, we attribute its inferior performance to the unsatisfactory result of back-translation augmentation on the e-commerce product data and the poor convergence of the model on the Books dataset. While we do not incorporate data augmentation methods into our framework, LTRN effectively learns from unlabeled data. Note that EDA data augmentation can also be added to our framework to further boost the model performance.

Comparing our model to the supervised reference model, the performance difference is around 8% on the Books dataset and less than 2% on the Amazon dataset, which is quite impressive given that the supervised model has 800× more labeled data on the Books dataset and 20× more on the Amazon dataset.

Table 4.2: Evaluation Results on Two Datasets. Methods are grouped into *supervised*, *text-based*, *network-based*, and *ours*. Underline marks the best score within each group.

	Amazon		**Books**	
Methods	Micro-F_1	Macro-F_1	Micro-F_1	Macro-F_1
BERT-supervised [10]	**0.938**	**0.921**	**0.853**	**0.855**
BERT-seed [10]	0.679	0.660	0.448	0.439
BERT-self-train	0.751	0.733	<u>0.599</u>	<u>0.611</u>
VAT [104]	0.336	0.321	0.305	0.265
EDA [85]	<u>0.793</u>	<u>0.781</u>	0.472	0.487
UDA [84]	0.748	0.711	0.434	0.414
WeSTClass [86]	0.337	0.315	0.387	0.404
CANE [88]	0.218	0.136	0.283	0.262
MetaCat [105]	0.389	0.374	0.419	0.437
Text-GCN [96]	0.684	<u>0.674</u>	0.505	0.494
Text-ING [106]	0.456	0.438	0.483	0.485
PPRGo-seed [100]	0.647	0.601	0.604	0.612
PPRGo-self-train	<u>0.687</u>	0.659	<u>0.691</u>	<u>0.697</u>
LTRN	**0.921**	**0.905**	**0.774**	**0.778**

4.4.5 Case Studies and Ablation Studies

In this section, we present case studies and ablation studies to scrutinize our proposed framework. Specifically, we aim to answer the following research questions: (1) How do the text and network modules mutually enhance each other through joint training? (2) Is our GNN design helpful for text-rich network modeling?

Different High-Confidence Predictions from Network and Text Modules. In the first case study, we show that the GNN model from the network module and the BERT model from the text module have different inductive biases, thus making different high confident predictions. We take high confident predictions T_1, T_2 from both models, as shown in Line 7,8 in Algorithm 4.1. Next, among the most confident predictions T_2 from GNN, we select those examples with **least** BERT confidence. Similarly, we also select examples where BERT is most confident, but GNN is least confident. The chosen examples show where one model does very well while the other model does poorly. We conduct this case study on the Amazon dataset. The examples are chosen from the very first iteration of the framework.

The results are shown in Table 4.3. In the first two rows, we show two products where GNN performs well, but BERT does not. The first product is a hidden safe disguised as a book, and the second product is a life vest. These two products are quite challenging judging from textual descriptions only. The first product has wordings similar to a book, and the

Table 4.3: Different High-Confidence Predictions from Network and Text Modules.

Document	Category	Prediction
BlueDot Trading Dictionary Secret Book Hidden Safe with Key Lock,...	Safe	BERT: (Writing Paper, 0.031), (Self Stick Note, 0.027) GNN: (Safe, 0.917)
Puoyis Toddler Kids Swim Life Vest, Girls and Boys Swim Float Jacket...	Water Flotation Device	BERT: (Dress, 0.044), (Incontinence Protector, 0.041) GNN: (Water Flotation Device, 0.987)
Logitech 910-000253 VX Nano Cordless Laser Mouse	Input Mouse	BERT: (Input Mouse, 0.907) GNN: (Input Mouse, 0.154), (Keyboard, 0.010)
PDP 048-082-NA-CM05 Stealth Series Wired Controller for Xbox One, Xbox One X	Game Controller	BERT: (Game Controller, 0.905) GNN: (Game Console, 0.149), (Keyboard, 0.125)

second product mentions words like "kids", "jacket" that are commonly seen in apparel products. From the BERT model prediction, we observe that it confuses the first product with paper, notes, and confuses the second product with dresses, incontinence protectors. The GNN model, on the other hand, successfully classifies them due to the information from the text-rich network structure, such as their brand and manufacturers.

The third and fourth row contains two examples where BERT does well, but GNN does not. The GNN cannot do well in those cases because products's neighbors come from a different category, and may mislead the GNN model. Starting from the brands "Logitech", "PDP", we find other peripheral devices of different categories; similarly, the key phrases, e.g., "Xbox One", "Xbox One X", connects the game controller product to game consoles through the network structure. Indeed, from the GNN predictions, we see that it confuses these products with "keyboard" and "game console".

The examples in the case study show that two modules have different strengths in the beginning iteration, and that there is an opportunity for one model to learn from the other model.

Per Category Performance Change Over Iterations. After observing that the network module and the text module learn to make different predictions in the first iteration, we are

Table 4.4: Performance of LTRN Model Ablations.

	Amazon		Books	
Methods	Micro-F_1	Macro-F_1	Micro-F_1	Macro-F_1
LTRN	0.921	0.905	0.774	0.778
LTRN (GraphSAGE)	0.898	0.871	0.756	0.759
LTRN (no-feat-share)	0.872	0.850	0.748	0.760

interested in whether both modules can mutually enhance each other through joint training. Therefore, we present a second case study on the per-category performance change of BERT and GNN models.

As shown in Figure 4.3, we present BERT and GNN model performance on different categories. We show results at the end of the first and the last framework iteration. In each subfigure, the categories are ordered according to the performance difference of BERT and GNN. From left to right, $\text{F1}_{\text{BERT}} - \text{F1}_{\text{GNN}}$ goes from largest to smallest. On the Amazon dataset, we select eight categories where two models have the most performance difference at the first iteration.

We can see that, in the beginning, the two models have different strengths in different categories. While in the end, the performance gaps of the two models are significantly narrowed, and the overall performance on these categories are much higher. Judging from the results, we can conclude that both modules are enhancing each other through joint training.

Effect of GNN Neighbor Sampling and Attention. We give two cases on the Amazon dataset to show how our proposed GNN architecture can select label-discriminative neighbors from mixed neighborhoods. As shown in Table 4.5, we compare direct neighbors of two products to the GNN re-ranked neighbors. The ordering of direct neighbors is according to the edge weight in the network, while the ordering of GNN neighbors is by $\alpha_{i,j}P_{i,j}$, according to Eqn. (4.4) where the attention values are from our fully trained GNN model. As we can see, in the two given cases, direct neighbors of the products are quite mixed, but our GNN model can rank meaningful neighbors higher after fully trained.

Ablation Study. We present two ablations of our model in Table 4.4 in order to verify the effectiveness of our proposed GNN architecture design and the feature sharing mechanism in joint training.

In the second row of Table 4.4, we replace our proposed GNN with a GraphSAGE model [92]. Unlike our model, GraphSAGE randomly samples node neighbors, and aggregates their features with a neural network model. We use the GCN variant of the GraphSAGE model, which does a weighted combination of neighbor features using normalized graph

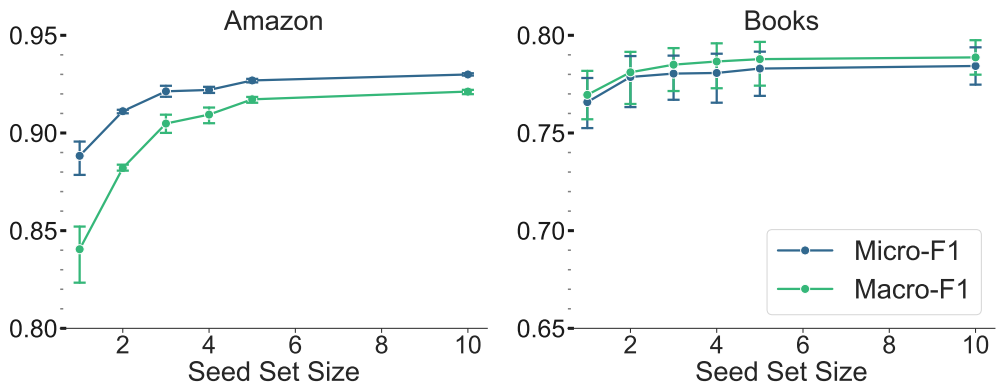


Figure 4.4: LTRN performance w.r.t #seed documents.

Laplacian as weights. On both datasets, we see a performance drop. We attribute the inferior performance of the model ablation to the lower quality pseudo labels generated by GraphSAGE compared to our proposed GNN architecture.

In the last row of Table 4.4, we remove the feature sharing mechanism and use the fixed unsupervised BERT embedding as GNN node features throughout the joint training. We can see a performance drop compared to LTRN. Without the help of updated BERT embedding, the GNN model produces slightly worse pseudo labels in later iterations and affects the overall categorization performance.

4.4.6 Seed Document Set Analysis

In this section, we aim to answer two research questions: (1) How does LTRN's performance change with respect to the number of seed documents? (2)How does the selection of labeled seed document impact model performance? In Figure 4.4, we present different runs of our LTRN model using different number of seed documents. We run 4 separate experiments for each seed set size. We use a different random selection of seed documents in each run.

As shown in Figure 4.4, the performance of the LTRN model generally increases when the number of labeled seed documents increases. The trend is more evident on the Amazon dataset than the Books dataset. It is also worth noting that when given only 1 seed example per category, the model already performs quite well on both datasets. Although on the Amazon dataset, we see a bigger gap between the Micro-F1 score and the Macro-F1 score when the number of seed documents is small, suggesting that the model's performance has a larger difference between categories in the most extreme minimal supervision setting.

We present the model's performance variance through error bars in the line plots. In general, there are two major sources of model variance: the selection of seed document as

Table 4.5: Ranked Neighbors after Network Learning vs. Direct Neighbors. × indicates neighbors whose category do not match with the given item.

GNN Ranked Neighbor	Direct Neighbor
Item: Bico Red & Blue Christmas Gnome Salt & Pepper Shaker Set...	
Bico Airtight Salt & Pepr. Shaker	Bico Airtight Salt & Pepr. Shaker
WL Disney Salt & Pepper Shaker...	× Garlic Press, S.S. Mincer
WL Ceramic Salt & Pepper Shaker...	× YF Chopper Crusher Machine
Item: Fortem Ratchet Tie Down Straps, 4x 15ft...	
Rhino Ratchet Straps 4pk	× Bovke Sci. Calculator Case
Rhino Ratchet Straps Heavy Duty	× Houseables Dog Tunnel
Autofonder 4 pk ratchet tie down	× Zomei Portable Tripod

input and the randomness in the model itself. The result shown in Figure 4.4 is a combination of both sources. Nevertheless, we can see that the model performance variance is less than ∼3% on two datasets. The model has a smaller variance on the Amazon dataset, and the variance gap becomes smaller as the number of seed documents increases. While on the Books dataset, the model has a larger variance, and the variance stays consistent for up to 10 seed documents per category.

4.5 RELATED WORK

Recent studies have shown increasing interest in training a good categorization model with little human effort. *Semi-supervised learning* aims to leverage both labeled and unlabeled data to boost a model's performance. *Weakly-supervised learning* incorporates other forms of supervision, such as giving seed words for each category [86, 91] or using only label surface names for classification [86, 107]. *Few-shot learning* focuses on model generalization, where the model is trained with a diverse set of categories and tested on new categories with a few examples [108, 109, 110]. Our minimally supervised setting resembles semi-supervised learning in extreme cases, where the number of labeled examples for each category is no more than 3.

Pseudo Labeling Methods. Combining labeled and unlabeled data for model training, pseudo labeling methods try to assign "pseudo labels" for unlabeled examples and incorporate them into model training. Self-training [111, 112] is arguably the most common strategy for pseudo label training. In each iteration of the algorithm, high confident predictions from the

model on the unlabeled dataset are added to the training set. Recent work improves self-training by assigning weights to pseudo labeled examples [87] with confident learning. Other extensions of self-training have explored using multiple models for pseudo label generation, e.g., co-training [113], tri-training [114], democratic co-learning [115]. Besides techniques that work on the input space, methods that incorporate pseudo examples by perturbation on the feature space have also achieved success [104]. Although our method adopts the pseudo labeling framework, we jointly train two models on both text and network data, as opposed to most prior works that only employ text data.

Data Augmentation Methods. This line of work augments the existing data examples to enlarge the training set. Recently, people have found data augmentation to work well with deep neural networks [84, 85, 116, 117]. Wei et al. [85] perform four simple operations: synonym replacement, random insertion, random swap, and random deletion. The augmented dataset consistently boosted the performance of deep learning-based text classifiers. Xie et al. [84] use back translation and TF-IDF word replacement to augment both labeled and unlabeled data. They leverage a joint loss function to tie both parts together. While our method does not use data augmentation, it is compatible with popular text augmentation techniques and could incorporate them into training.

Another related thread to our framework is graph semi-supervised learning. Graph neural networks (GNNs) bring deep learning to graph-structured data by message passing on graphs [97, 118]. Most GNN models assume fixed vectorized node features, while in our work, raw text coming with the nodes plays a predominant role in categorization. Pioneer studies bringing text and network together typically use a text-centric model to maximize edge probabilities in a graph [88, 89] or use a network-centric model to learn node representations [105] or use a fixed random walk process to propagate label on the graph [90, 91]. In this paper, we bring text and graph learning together for minimal supervision.

4.6 CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a minimally supervised text categorization model by learning on text-rich networks. We leverage a BERT model for raw text understanding and proposed a novel GNN model to model label-discriminative signal in the text-rich network structure. We jointly train the BERT model with the GNN model with co-training and feature sharing. With only a few user given seed examples, the model shows competitive performance even when compared to a supervised model.

In the future, we would like to explore models that can capture heterogeneous type information in the text-rich network. We would also like to incorporate label type taxonomy into our framework.

Chapter 5: CONCLUSION AND FUTURE WORK

This dissertation proposed text-rich network approaches for weakly-supervised text mining, pioneering systematic study of the synergy between text and structured entities in semi-structured real-world corpora. We contributed to both main steps of text-rich network-based text mining: construction and mining.

For text-rich network construction and consolidation, we introduced novel methods for open-world attribute mining, extracting both new attribute types and values from raw text to complement existing structure. For text-rich network mining, we demonstrated how text-rich networks enable training strong pre-trained language models and reduce human labeling effort in text classification.

During the writing of this thesis, the field of text mining and natural language processing experienced a significant breakthrough with the rise of large language models [58, 119] (LLMs). LLMs have revolutionized approaches to classical NLP problems like question answering, language generation, and information extraction. From a methodology perspective, LLMs could potentially enhance some of the works proposed in this thesis. For example, in chapter 2's two-step open-world attribute mining framework, LLMs may enable extending the candidate generation step beyond e-commerce product titles to more free-form text.

However, the fundamental principles of our proposed works remain relevant even with LLM advancements. Our studies illustrate the complementary nature of textual and network models. More powerful language models could refresh the smaller pre-trained models used in our works and further advance our frameworks. Text-rich networks are poised to play an important ongoing role in the age of LLMs.

Several future research directions emerge from this work. First, our text-rich networks could enhance LLMs as tools, with their structured format advancing retrieval-augmented LLMs through stronger, richer retrieval sources. Second, with progress in multimodal foundation models, an ambitious but rewarding direction is to explore networks as a modality for LLMs. The natural integration of unstructured text and semi-structured entities in our text-rich networks could serve as a data backbone for large-scale model training.

In conclusion, this dissertation has demonstrated the power and potential of text-rich network approaches for weakly-supervised text mining. By pioneering methods for constructing text-rich networks from semi-structured corpora and leveraging them for tasks like language model pre-training and labeling-efficient text classification, we have laid a foundation for harnessing the synergy of textual and network models. As large language models continue to reshape the NLP landscape, text-rich networks are well-positioned to play an enabling

role - enhancing LLMs through richer retrieval and potentially even serving as a modality for multimodal foundation models. Overall, this work opens up exciting avenues for future research at the intersection of text mining, network science, and large language models.

Appendix A: DETAILS AND ADDITIONAL RESULTS

A.1 TWHIN-BERT DETAILS AND ADDITIONAL RESULTS

A.1.1 Distribution of Languages in Training Dataset

Figure A.1 shows the distribution of languages in our pre-training dataset. Some languages with different variations (e.g., Hindi and Hindi Romanized) are represented with the same ISO language code. We run fastText [120] language identification model `lid.176.bin`[11] to detect languages.

We deem a language "high-resource" if we have more than 10^8 Tweets during pre-training *after* frequency-based re-sampling (Section 3.2.3); "mid-resource" if we have more than 10^7 and less than 10^8 Tweets; "low-resource" if we have less than 10^7 Tweets.

A.1.2 Hyperparameters for Pre-training and Fine-Tuning

Table A.1 shows the pre-training hyperparameters. The model architecture and hyperparameters not shown in the table are the same as RoBERTa [44].

Table A.2 shows the hyperparameters for classification fine-tuning. We do hyperparameter selection on the development datasets and share the same set of hyperparameters for the base models, as we find them to perform well with this setting. The weight decay for base models is set to zero. A different set of hyperparameters were necessary for the large model because it behaves differently from the base models in terms of convergence.

A.1.3 Evaluation Metrics for External Classification Benchmarks

The recommended evaluation metrics that we report in Table 3.6 are as follows. Average recall for ASAD, SemEval 2017 datasets; Macro-F1 for SemEval 2018 English and Spanish datasets; Accuracy for COVID-JA, SemEval 2020 datasets.

A.1.4 Engagement Prediction Results on Additional Languages

Table 3.8 shows the engagement prediction results on all available evaluation languages. Some languages have more examples than other languages due to data availability.

[11] https://fasttext.cc/docs/en/language-identification.html

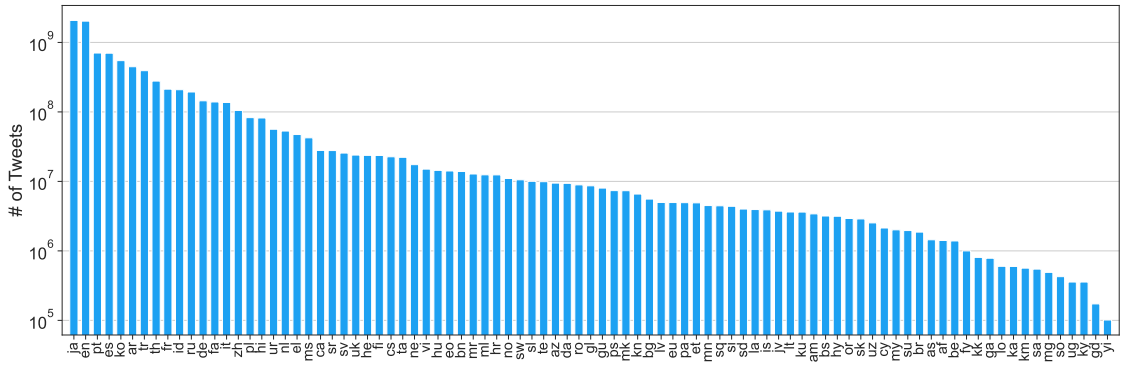


Figure A.1: The number of Tweets in the pre-training dataset for each language. Languages are marked by ISO language codes.

A.1.5 Hashtag Prediction Results on Additional Languages

Table 3.9 shows the hashtag prediction results on all available evaluation languages. A small number of languages have less examples than shown in Table 3.4 due to data availability. The Russian language is not evaluated as the XLM-T baseline fails on some Russian characters in our dataset.

Table A.1: Hyperparameters for pre-training TwHIN-BERT.

Hyperparameter	**TwHIN-BERT-base**	**TwHIN-BERT-large**
Max sequence length	128	128
Precision	BF16	BF16
Stage 1: MLM		
Total batch size	6K	8K
Gradient accumulation steps	1	4
Peak learning rate	2e-4	2e-4
Warmup steps	30K	30K
Total steps	500K	500K
Stage 2: MLM + Social		
Total batch size	6K	6K
Gradient checkpointing	No	Yes
Peak learning rate	1e-4	1e-4
Warmup steps	30K	30K
Total steps	500K	500K
Contrastive projection head	[768, 768]	[1024, 512]
Contrastive loss temperature	0.1	0.1
Loss balancing λ	0.05	0.05

Table A.2: Hyperparameters for fine-tuning TwHIN-BERT and the baselines for classification.

Hyperparameter	**Hashtag**	**SE2017**	**SE2018**	**ASAD**	**COVID-JA**	**SE2020**
Base models						
Learning rate	4e-5	4e-5	1e-5	1e-5	2e-5	2e-5
Batch size	128	128	128	128	128	128
TwHIN-BERT-large						
Learning rate	2e-5	2e-5	1e-5	1e-5	1e-5	1e-5
Weight decay	0	0	5e-4	5e-4	0	5e-4
Batch size	128	128	128	128	128	128

REFERENCES

[1] K. D. Bollacker, C. Evans, P. K. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, J. T. Wang, Ed. ACM, 2008. [Online]. Available: https://doi.org/10.1145/1376616.1376746 pp. 1247–1250.

[2] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: a web-scale approach to probabilistic knowledge fusion," in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, S. A. Macskassy, C. Perlich, J. Leskovec, W. Wang, and R. Ghani, Eds. ACM, 2014. [Online]. Available: https://doi.org/10.1145/2623330.2623623 pp. 601–610.

[3] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *Proc. VLDB Endow.*, vol. 4, no. 11, pp. 992–1003, 2011. [Online]. Available: http://www.vldb.org/pvldb/vol4/p992-sun.pdf

[4] X. L. Dong, X. He, A. Kan, X. Li, Y. Liang, J. Ma, Y. E. Xu, C. Zhang, T. Zhao, G. B. Saldana, S. Deshpande, A. M. Manduca, J. Ren, S. P. Singh, F. Xiao, H. Chang, G. Karamanolakis, Y. Mao, Y. Wang, C. Faloutsos, A. McCallum, and J. Han, "Autoknow: Self-driving knowledge collection for products of thousands of types," in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, Eds. ACM, 2020. [Online]. Available: https://doi.org/10.1145/3394486.3403323 pp. 2724–2734.

[5] X. Ren, A. El-Kishky, C. Wang, F. Tao, C. R. Voss, and J. Han, "Clustype: Effective entity recognition and typing by relation phrase-based clustering," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, L. Cao, C. Zhang, T. Joachims, G. I. Webb, D. D. Margineantu, and G. Williams, Eds. ACM, 2015. [Online]. Available: https://doi.org/10.1145/2783258.2783362 pp. 995–1004.

[6] G. Zheng, S. Mukherjee, X. L. Dong, and F. Li, "Opentag: Open attribute value extraction from product profiles," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3219819.3219839 p. 1049–1058.

[7] H. Xu, W. Wang, X. Mao, X. Jiang, and M. Lan, "Scaling up open tagging from tens to thousands: Comprehension empowered attribute value extraction from product title," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019. [Online]. Available: https://aclanthology.org/P19-1514 pp. 5214–5223.

[8] J. Yan, N. Zalmout, Y. Liang, C. Grant, X. Ren, and X. L. Dong, "AdaTag: Multi-attribute value extraction from product profiles with adaptive decoding," 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)*. Online: Association for Computational Linguistics, Aug. 2021. [Online]. Available: https://aclanthology.org/2021.acl-long.362 pp. 4694–4705.

[9] Q. Wang, L. Yang, B. Kanagal, S. Sanghai, D. Sivakumar, B. Shu, Z. Yu, and J. Elsas, "Learning to extract attribute value from product via question answering: A multi-task approach," in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, Eds. ACM, 2020. [Online]. Available: https://doi.org/10.1145/3394486.3403047 pp. 47–55.

[10] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds. Association for Computational Linguistics, 2019. [Online]. Available: https://doi.org/10.18653/v1/n19-1423 pp. 4171–4186.

[11] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, "spacy: Industrial-strength natural language processing in python," 2020, spacy.

[12] A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf, "Flair: An easy-to-use framework for state-of-the-art nlp," in *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, 2019, pp. 54–59.

[13] X. Gu, Z. Wang, Z. Bi, Y. Meng, L. Liu, J. Han, and J. Shang, "Ucphrase: Unsupervised context-aware quality phrase tagging," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, ser. KDD '21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3447548.3467397 p. 478–486.

[14] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, 1996, p. 226–231.

[15] T. Kim, J. Choi, D. Edmiston, and S. Lee, "Are pre-trained language models aware of phrases? simple but strong baselines for grammar induction," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. [Online]. Available: https://openreview.net/forum?id=H1xPR3NtPB

[16] Z. Wu, Y. Chen, B. Kao, and Q. Liu, "Perturbed masking: Parameter-free probing for analyzing and interpreting BERT," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020. [Online]. Available: https://aclanthology.org/2020.acl-main.383 pp. 4166–4176.

[17] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," 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)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019. [Online]. Available: https://aclanthology.org/D19-1410 pp. 3982–3992.

[18] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*. IEEE Computer Society, 2015. [Online]. Available: https://doi.org/10.1109/CVPR.2015.7298682 pp. 815–823.

[19] M. Schultz and T. Joachims, "Learning a distance metric from relative comparisons," in *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada]*, S. Thrun, L. K. Saul, and B. Schölkopf, Eds. MIT Press, 2003. [Online]. Available: https://proceedings.neurips.cc/paper/2003/hash/d3b1fb02964aa64e257f9f26a31f72cf-Abstract.html pp. 41–48.

[20] G. Chechik, V. Sharma, U. Shalit, and S. Bengio, "Large scale online learning of image similarity through ranking," *J. Mach. Learn. Res.*, vol. 11, pp. 1109–1135, 2010. [Online]. Available: https://dl.acm.org/doi/10.5555/1756006.1756042

[21] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020. [Online]. Available: https://proceedings.mlr.press/v119/chen20j.html pp. 1597–1607.

[22] J. Shang, J. Liu, M. Jiang, X. Ren, C. R. Voss, and J. Han, "Automated phrase mining from massive text corpora," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 10, pp. 1825–1837, 2018.

[23] H. Zhang, H. Xu, T.-E. Lin, and R. Lyu, "Discovering new intents with deep aligned clustering," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 16, pp. 14 365–14 373, May 2021. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/17689

[24] M. J. Zaki and W. Meira, *Data Mining and Machine Learning: Fundamental Concepts and Algorithms*, 2nd ed. Cambridge University Press, 2020. [Online]. Available: https://dataminingbook.info/book_html/

[25] D. Putthividhya and J. Hu, "Bootstrapped named entity recognition for product attribute extraction," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, July 2011. [Online]. Available: https://aclanthology.org/D11-1144 pp. 1557–1567.

[26] H. Xu, B. Liu, L. Shu, and P. Yu, "Open-world learning and application to product classification," in *The World Wide Web Conference*, ser. WWW '19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3308558.3313644 p. 3413–3419.

[27] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016. [Online]. Available: https://proceedings.mlr.press/v48/xieb16.html pp. 478–487.

[28] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, Eds. Association for Computational Linguistics, 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.747 pp. 8440–8451.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html pp. 5998–6008.

[30] D. Q. Nguyen, T. Vu, and A. T. Nguyen, "Bertweet: A pre-trained language model for english tweets," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, Q. Liu and D. Schlangen, Eds. Association for Computational Linguistics, 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.emnlp-demos.2 pp. 9–14.

[31] F. Barbieri, L. E. Anke, and J. Camacho-Collados, "XLM-T: A multilingual language model toolkit for twitter," *CoRR*, vol. abs/2104.12250, 2021. [Online]. Available: https://arxiv.org/abs/2104.12250

[32] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Y. Guo and F. Farooq, Eds. ACM, 2018. [Online]. Available: https://doi.org/10.1145/3219819.3219890 pp. 974–983.

[33] J. Cheng, L. A. Adamic, P. A. Dow, J. M. Kleinberg, and J. Leskovec, "Can cascades be predicted?" in *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014*, C. Chung, A. Z. Broder, K. Shim, and T. Suel, Eds. ACM, 2014. [Online]. Available: https://doi.org/10.1145/2566486.2567997 pp. 925–936.

[34] A. Sankar, X. Zhang, A. Krishnan, and J. Han, "Inf-vae: A variational autoencoder framework to integrate homophily and influence in diffusion prediction," in *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, J. Caverlee, X. B. Hu, M. Lalmas, and W. Wang, Eds. ACM, 2020. [Online]. Available: https://doi.org/10.1145/3336191.3371811 pp. 510–518.

[35] A. El-Kishky, T. Markovich, S. Park, C. Verma, B. Kim, R. Eskander, Y. Malkov, F. Portman, S. Samaniego, Y. Xiao, and A. Haghighi, "Twhin: Embedding the twitter heterogeneous information network for personalized recommendation," in *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, A. Zhang and H. Rangwala, Eds. ACM, 2022. [Online]. Available: https://doi.org/10.1145/3534678.3539080 pp. 2842–2850.

[36] A. El-Kishky, T. Markovich, K. Leung, F. Portman, A. Haghighi, and Y. Xiao, "knn-embed: Locally smoothed embedding mixtures for multi-interest candidate retrieval," *CoRR*, vol. abs/2205.06205, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2205.06205

[37] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, Eds., 2013. [Online]. Available: https://proceedings.neurips.cc/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html pp. 3111–3119.

[38] Y. Goldberg and O. Levy, "word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method," *CoRR*, vol. abs/1402.3722, 2014. [Online]. Available: http://arxiv.org/abs/1402.3722

[39] A. Lerer, L. Wu, J. Shen, T. Lacroix, L. Wehrstedt, A. Bose, and A. Peysakhovich, "Pytorch-biggraph: A large scale graph embedding system," in *Proceedings of the SysML Conference 2019 (SysML 2019), Stanford, CA, USA, March 31 - April 2, 2019*, A. Talwalkar, V. Smith, and M. Zaharia, Eds. mlsys.org, 2019. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2019/hash/1eb34d662b67a14e3511d0dfd78669be-Abstract.html

[40] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, Eds., 2013. [Online]. Available: https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html pp. 2787–2795.

[41] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 535–547, 2021. [Online]. Available: https://doi.org/10.1109/TBDATA.2019.2921572

[42] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 117–128, 2011. [Online]. Available: https://doi.org/10.1109/TPAMI.2010.57

[43] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 2020. [Online]. Available: http://proceedings.mlr.press/v119/chen20j.html pp. 1597–1607.

[44] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019. [Online]. Available: http://arxiv.org/abs/1907.11692

[45] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, A. Gangemi, S. Leonardi, and A. Panconesi, Eds. ACM, 2015. [Online]. Available: https://doi.org/10.1145/2736277.2741093 pp. 1067–1077.

[46] F. Mireshghallah, N. Vogler, J. He, O. Florez, A. El-Kishky, and T. Berg-Kirkpatrick, "Non-parametric temporal adaptation for social media topic classification," *CoRR*, vol. abs/2209.05706, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2209.05706

[47] S. Rosenthal, N. Farra, and P. Nakov, "Semeval-2017 task 4: Sentiment analysis in twitter," *CoRR*, vol. abs/1912.00741, 2019. [Online]. Available: http://arxiv.org/abs/1912.00741

[48] B. Alharbi, H. Alamro, M. Alshehri, Z. Khayyat, M. Kalkatawi, I. I. Jaber, and X. Zhang, "ASAD: A twitter-based benchmark arabic sentiment analysis dataset," *CoRR*, vol. abs/2011.00578, 2020. [Online]. Available: https://arxiv.org/abs/2011.00578

[49] P. Patwa, G. Aguilar, S. Kar, S. Pandey, S. PYKL, B. Gambäck, T. Chakraborty, T. Solorio, and A. Das, "Semeval-2020 task 9: Overview of sentiment analysis of code-mixed tweets," in *Proceedings of the Fourteenth Workshop on Semantic Evaluation, SemEval@COLING 2020, Barcelona (online), December 12-13, 2020*, A. Herbelot, X. Zhu, A. Palmer, N. Schneider, J. May, and E. Shutova, Eds. International Committee for Computational Linguistics, 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.semeval-1.100 pp. 774–790.

[50] F. Barbieri, J. Camacho-Collados, F. Ronzano, L. Espinosa-Anke, M. Ballesteros, V. Basile, V. Patti, and H. Saggion, "SemEval 2018 task 2: Multilingual emoji prediction," in *Proceedings of The 12th International Workshop on Semantic Evaluation*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018. [Online]. Available: https://aclanthology.org/S18-1003 pp. 24–33.

[51] Y. Suzuki, "Filtering method for twitter streaming data using human-in-the-loop machine learning," *J. Inf. Process.*, vol. 27, pp. 404–410, 2019. [Online]. Available: https://doi.org/10.2197/ipsjjip.27.404

[52] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever et al., "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[53] A. Conneau and G. Lample, "Cross-lingual language model pretraining," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/c04c19c2c2474dbf5f7ac4372c5b9af1-Abstract.html pp. 7057–7067.

[54] P. Rust, J. Pfeiffer, I. Vulic, S. Ruder, and I. Gurevych, "How good is your tokenizer? on the monolingual performance of multilingual language models," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, C. Zong, F. Xia, W. Li, and R. Navigli, Eds. Association for Computational Linguistics, 2021. [Online]. Available: https://doi.org/10.18653/v1/2021.acl-long.243 pp. 3118–3135.

[55] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel, "mt5: A massively multilingual pre-trained text-to-text transformer," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, Eds. Association for Computational Linguistics, 2021. [Online]. Available: https://doi.org/10.18653/v1/2021.naacl-main.41 pp. 483–498.

[56] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, M. A. Walker, H. Ji, and A. Stent, Eds. Association for Computational Linguistics, 2018. [Online]. Available: https://doi.org/10.18653/v1/n18-1202 pp. 2227–2237.

[57] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/dc6a7e655d7e5840e66733e9ee67cc69-Abstract.html pp. 5754–5764.

[58] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[59] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, pp. 140:1–140:67, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-074.html

[60] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro, "Megatron-lm: Training multi-billion parameter language models using model parallelism," *CoRR*, vol. abs/1909.08053, 2019. [Online]. Available: http://arxiv.org/abs/1909.08053

[61] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," *CoRR*, vol. abs/2101.03961, 2021. [Online]. Available: https://arxiv.org/abs/2101.03961

[62] K. Clark, M. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: pre-training text encoders as discriminators rather than generators," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. [Online]. Available: https://openreview.net/forum?id=r1xMH1BtvB

[63] Y. Meng, C. Xiong, P. Bajaj, S. Tiwary, P. Bennett, J. Han, and X. Song, "COCO-LM: correcting and contrasting text sequences for language model pretraining," in *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/c2c2a04512b35d13102459f8784f1a2d-Abstract.html pp. 23 102–23 114.

[64] D. Loureiro, F. Barbieri, L. Neves, L. E. Anke, and J. Camacho-Collados, "Timelms: Diachronic language models from twitter," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL 2022 - System Demonstrations, Dublin, Ireland, May 22-27, 2022*, V. Basile, Z. Kozareva, and S. Stajner, Eds. Association for Computational Linguistics, 2022. [Online]. Available: https://doi.org/10.18653/v1/2022.acl-demo.25 pp. 251–260.

[65] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, "ERNIE: enhanced language representation with informative entities," in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, A. Korhonen, D. R. Traum, and L. Màrquez, Eds. Association for Computational Linguistics, 2019. [Online]. Available: https://doi.org/10.18653/v1/p19-1139 pp. 1441–1451.

[66] W. Liu, P. Zhou, Z. Zhao, Z. Wang, Q. Ju, H. Deng, and P. Wang, "K-bert: Enabling language representation with knowledge graph," in *AAAI*, 2020.

[67] X. Liu, D. Yin, J. Zheng, X. Zhang, P. Zhang, H. Yang, Y. Dong, and J. Tang, "OAG-BERT: towards a unified backbone language model for academic knowledge services," in *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, A. Zhang and H. Rangwala, Eds. ACM, 2022. [Online]. Available: https://doi.org/10.1145/3534678.3539210 pp. 3418–3428.

[68] M. Yasunaga, J. Leskovec, and P. Liang, "LinkBERT: Pretraining language models with document links," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022. [Online]. Available: https://aclanthology.org/2022.acl-long.551 pp. 8003–8016.

[69] A. El-Kishky, M. Bronstein, Y. Xiao, and A. Haghighi, "Graph-based representation learning for web-scale recommender systems," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 4784–4785.

[70] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, S. A. Macskassy, C. Perlich, J. Leskovec, W. Wang, and R. Ghani, Eds. ACM, 2014. [Online]. Available: https://doi.org/10.1145/2623330.2623732 pp. 701–710.

[71] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, Eds. ACM, 2016. [Online]. Available: https://doi.org/10.1145/2939672.2939754 pp. 855–864.

[72] Y. Sun and J. Han, "Mining heterogeneous information networks: a structural analysis approach," *SIGKDD Explor.*, vol. 14, no. 2, pp. 20–28, 2012. [Online]. Available: https://doi.org/10.1145/2481244.2481248

[73] S. Chang, W. Han, J. Tang, G. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, L. Cao, C. Zhang, T. Joachims, G. I. Webb, D. D. Margineantu, and G. Williams, Eds. ACM, 2015. [Online]. Available: https://doi.org/10.1145/2783258.2783296 pp. 119–128.

[74] J. Tang, M. Qu, and Q. Mei, "PTE: predictive text embedding through large-scale heterogeneous text networks," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, L. Cao, C. Zhang, T. Joachims, G. I. Webb, D. D. Margineantu, and G. Williams, Eds. ACM, 2015. [Online]. Available: https://doi.org/10.1145/2783258.2783307 pp. 1165–1174.

[75] L. Xu, X. Wei, J. Cao, and P. S. Yu, "Embedding of embedding (EOE): joint embedding for coupled heterogeneous networks," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*, M. de Rijke, M. Shokouhi, A. Tomkins, and M. Zhang, Eds. ACM, 2017. [Online]. Available: https://doi.org/10.1145/3018661.3018723 pp. 741–749.

[76] T. Chen and Y. Sun, "Task-guided and path-augmented heterogeneous network embedding for author identification," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*, M. de Rijke, M. Shokouhi, A. Tomkins, and M. Zhang, Eds. ACM, 2017. [Online]. Available: https://doi.org/10.1145/3018661.3018735 pp. 295–304.

[77] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. ACM, 2017. [Online]. Available: https://doi.org/10.1145/3097983.3098036 pp. 135–144.

[78] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, 2017. [Online]. Available: https://doi.org/10.1109/TKDE.2017.2754499

[79] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, ser. JMLR Workshop and Conference Proceedings, M. Balcan and K. Q. Weinberger, Eds., vol. 48. JMLR.org, 2016. [Online]. Available: http://proceedings.mlr.press/v48/trouillon16.html pp. 2071–2080.

[80] C. Sun, N. Rampalli, F. Yang, and A. Doan, "Chimera: Large-scale classification using machine learning, rules, and crowdsourcing," *Proc. VLDB Endow.*, vol. 7, no. 13, pp. 1529–1540, 2014. [Online]. Available: http://www.vldb.org/pvldb/vol7/p1529-sun.pdf

[81] H. Chen, J. Zhao, and D. Yin, "Fine-grained product categorization in e-commerce," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, W. Zhu, D. Tao, X. Cheng, P. Cui, E. A. Rundensteiner, D. Carmel, Q. He, and J. X. Yu, Eds. ACM, 2019. [Online]. Available: https://doi.org/10.1145/3357384.3358170 pp. 2349–2352.

[82] L. Song, P. Pan, K. Zhao, H. Yang, Y. Chen, Y. Zhang, Y. Xu, and R. Jin, "Large-scale training system for 100-million classification at alibaba," in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, Eds. ACM, 2020. [Online]. Available: https://doi.org/10.1145/3394486.3403342 pp. 2909–2930.

[83] K. Leetaru and P. A. Schrodt, "Gdelt: Global data on events, location, and tone, 1979–2012," in *ISA annual convention*, vol. 2, no. 4. Citeseer, 2013, pp. 1–49.

[84] Q. Xie, Z. Dai, E. H. Hovy, M. Luong, and Q. V. Le, "Unsupervised data augmentation," *CoRR*, vol. abs/1904.12848, 2019. [Online]. Available: http://arxiv.org/abs/1904.12848

[85] J. W. Wei and K. Zou, "EDA: easy data augmentation techniques for boosting performance on text classification tasks," 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 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 2019. [Online]. Available: https://doi.org/10.18653/v1/D19-1670 pp. 6381–6387.

[86] Y. Meng, J. Shen, C. Zhang, and J. Han, "Weakly-supervised neural text classification," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, A. Cuzzocrea, J. Allan, N. W. Paton, D. Srivastava, R. Agrawal, A. Z. Broder, M. J. Zaki, K. S. Candan, A. Labrinidis, A. Schuster, and H. Wang, Eds. ACM, 2018. [Online]. Available: https://doi.org/10.1145/3269206.3271737 pp. 983–992.

[87] X. Li, Q. Sun, Y. Liu, Q. Zhou, S. Zheng, T. Chua, and B. Schiele, "Learning to self-train for semi-supervised few-shot classification," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, 2019. [Online]. Available: http://papers.nips.cc/paper/9216-learning-to-self-train-for-semi-supervised-few-shot-classification pp. 10 276–10 286.

[88] C. Tu, H. Liu, Z. Liu, and M. Sun, "CANE: context-aware network embedding for relation modeling," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. Association for Computational Linguistics, 2017. [Online]. Available: https://doi.org/10.18653/v1/P17-1158 pp. 1722–1731.

[89] D. Shen, X. Zhang, R. Henao, and L. Carin, "Improved semantic-aware network embedding with fine-grained word alignment," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, 2018. [Online]. Available: https://doi.org/10.18653/v1/d18-1209 pp. 1829–1838.

[90] J. Shang, X. Zhang, L. Liu, S. Li, and J. Han, "Nettaxo: Automated topic taxonomy construction from text-rich network," in *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, Y. Huang, I. King, T. Liu, and M. van Steen, Eds. ACM / IW3C2, 2020. [Online]. Available: https://doi.org/10.1145/3366423.3380259 pp. 1908–1919.

[91] D. Mekala, X. Zhang, and J. Shang, "META: metadata-empowered weak supervision for text classification," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Association for Computational Linguistics, 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.emnlp-main.670 pp. 8351–8361.

[92] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 2017. [Online]. Available: http://papers.nips.cc/paper/6703-inductive-representation-learning-on-large-graphs pp. 1024–1034.

[93] J. Chen, T. Ma, and C. Xiao, "Fastgcn: Fast learning with graph convolutional networks via importance sampling," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=rytstxWAW

[94] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[95] J. Shang, J. Liu, M. Jiang, X. Ren, C. R. Voss, and J. Han, "Automated phrase mining from massive text corpora," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 10, pp. 1825–1837, 2018. [Online]. Available: https://doi.org/10.1109/TKDE.2018.2812203

[96] L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification," in *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2019. [Online]. Available: https://doi.org/10.1609/aaai.v33i01.33017370 pp. 7370–7377.

[97] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl

[98] T. H. Haveliwala, "Topic-sensitive pagerank," in *Proceedings of the Eleventh International World Wide Web Conference, WWW 2002, May 7-11, 2002, Honolulu, Hawaii, USA*. ACM, 2002. [Online]. Available: https://doi.org/10.1145/511446.511513 pp. 517–526.

[99] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=H1gL-2A9Ym

[100] A. Bojchevski, J. Klicpera, B. Perozzi, A. Kapoor, M. Blais, B. Rózemberczki, M. Lukasik, and S. Günnemann, "Scaling graph neural networks with approximate pagerank," in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*. ACM, 2020. [Online]. Available: https://dl.acm.org/doi/10.1145/3394486.3403296 pp. 2464–2473.

[101] H. Tong, C. Faloutsos, and J. Pan, "Fast random walk with restart and its applications," in *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), 18-22 December 2006, Hong Kong, China*. IEEE Computer Society, 2006. [Online]. Available: https://doi.org/10.1109/ICDM.2006.70 pp. 613–622.

[102] R. Andersen, F. R. K. Chung, and K. J. Lang, "Local graph partitioning using pagerank vectors," in *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*. IEEE Computer Society, 2006. [Online]. Available: https://doi.org/10.1109/FOCS.2006.44 pp. 475–486.

[103] M. Wan and J. J. McAuley, "Item recommendation on monotonic behavior chains," in *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*. ACM, 2018. [Online]. Available: https://doi.org/10.1145/3240323.3240369 pp. 86–94.

[104] T. Miyato, A. M. Dai, and I. J. Goodfellow, "Adversarial training methods for semi-supervised text classification," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=r1X3g2_xl

[105] Y. Zhang, Y. Meng, J. Huang, F. F. Xu, X. Wang, and J. Han, "Minimally supervised categorization of text with metadata," in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 2020. [Online]. Available: https://doi.org/10.1145/3397271.3401168 pp. 1231–1240.

[106] Y. Zhang, X. Yu, Z. Cui, S. Wu, Z. Wen, and L. Wang, "Every document owns its structure: Inductive text classification via graph neural networks," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, Eds. Association for Computational Linguistics, 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.31 pp. 334–339.

[107] Y. Meng, Y. Zhang, J. Huang, C. Xiong, H. Ji, C. Zhang, and J. Han, "Text classification using label names only: A language model self-training approach," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Association for Computational Linguistics, 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.emnlp-main.724 pp. 9006–9017.

[108] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, 2006. [Online]. Available: https://doi.org/10.1109/TPAMI.2006.79

[109] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, Eds., 2016. [Online]. Available: https://proceedings.neurips.cc/paper/2016/hash/90e1357833654983612fb05e3ec9148c-Abstract.html pp. 3630–3638.

[110] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/cb8da6767461f2812ae4290eac7cbc42-Abstract.html pp. 4077–4087.

[111] R. Mihalcea, "Co-training and self-training for word sense disambiguation," in *Proceedings of the Eighth Conference on Computational Natural Language Learning, CoNLL 2004, Held in cooperation with HLT-NAACL 2004, Boston, Massachusetts, USA, May 6-7, 2004*, H. T. Ng and E. Riloff, Eds. ACL, 2004. [Online]. Available: https://aclanthology.org/W04-2405/ pp. 33–40.

[112] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *33rd Annual Meeting of the Association for Computational Linguistics, 26-30 June 1995, MIT, Cambridge, Massachusetts, USA, Proceedings*. Morgan Kaufmann Publishers / ACL, 1995. [Online]. Available: https://www.aclweb.org/anthology/P95-1026/ pp. 189–196.

[113] A. Blum and T. M. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT 1998, Madison, Wisconsin, USA, July 24-26, 1998*, P. L. Bartlett and Y. Mansour, Eds. ACM, 1998. [Online]. Available: https://doi.org/10.1145/279943.279962 pp. 92–100.

[114] Z. Zhou and M. Li, "Tri-training: Exploiting unlabeled data using three classifiers," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1529–1541, 2005. [Online]. Available: https://doi.org/10.1109/TKDE.2005.186

[115] Y. Zhou and S. A. Goldman, "Democratic co-learning," in *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004), 15-17 November 2004, Boca Raton, FL, USA*. IEEE Computer Society, 2004. [Online]. Available: https://doi.org/10.1109/ICTAI.2004.48 pp. 594–602.

[116] S. Kobayashi, "Contextual augmentation: Data augmentation by words with paradigmatic relations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018. [Online]. Available: https://www.aclweb.org/anthology/N18-2072 pp. 452–457.

[117] S. Qiu, B. Xu, J. Zhang, Y. Wang, X. Shen, G. de Melo, C. Long, and X. Li, "Easyaug: An automatic textual data augmentation platform for classification tasks," in *Companion Proceedings of the Web Conference 2020*, ser. WWW '20. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3366424.3383552 p. 249–252.

[118] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 2017. [Online]. Available: http://proceedings.mlr.press/v70/gilmer17a.html pp. 1263–1272.

[119] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel, "Palm: Scaling language modeling with pathways," *J. Mach. Learn. Res.*, vol. 24, pp. 240:1–240:113, 2023. [Online]. Available: http://jmlr.org/papers/v24/22-1144.html

[120] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017. [Online]. Available: https://aclanthology.org/Q17-1010