Keyphrase Identification Using Minimal Labeled Data with Hierarchical Context and Transfer Learning

Goli, Rohan ; Hubig, Nina; Min, Hua; Gong, Yang; Sittig, Dean F.; Rennert, Lior; Robinson, David; Biondich, Paul; Wright, Adam; Nøhr, Christian Gradhandt; Law, Timothy; Faxvaag, Arild; Weaver, Aneesa ; Gimbel, Ronald; Jing, Xia

Published in:
medRxiv

DOI (link to publication from Publisher):
10.1101/2023.01.26.23285060

Publication date:
2023

Document Version
Early version, also known as pre-print

Keyphrase Identification Using Minimal Labeled Data with Hierarchical Context and Transfer Learning

Rohan Goli, MS[1]; Nina Hubig, PhD[1]; Hua Min, PhD[2]; Yang Gong, PhD[3]; Dean F. Sittig, PhD[3]; Lior Rennert, PhD[4]; David Robinson, MD[5]; Paul Biondich, MD[6]; Adam Wright, PhD[7]; Christian Nøhr, PhD[8]; Timothy Law, DO[9]; Arild Faxvaag, PhD[10]; Aneesa Weaver, BS[4]; Ronald Gimbel, PhD[4]; Xia Jing, PhD[4]

[1]School of Computing, College of Engineering, Computing and Applied Science, Clemson University, Clemson, SC, USA; [2]Department of Health Administration and Policy, College of Public Health, George Mason University, Fairfax, VA, USA; [3]School of Biomedical Informatics, The University of Texas Health Science Center at Houston, Houston, TX, USA; [4]Department of Public Health Sciences, College of Behavioral, Social, and Health Sciences, Clemson University, Clemson, SC, USA; [5]General Practitioner/Independent Consultant, Cumbria, UK; [6]Clem McDonald Biomedical Informatics Center, Regenstrief Institute, Department of Pediatrics, Indiana University School of Medicine, Indianapolis, IN, USA; [7]Vanderbilt University Medical Center, Nashville, TN, USA; [8]Department of Planning, Faculty of Engineering, Aalborg University, Aalborg, Denmark; [9]Ohio Musculoskeletal and Neurologic Institute, Ohio University, Athens, OH, USA; [10]Department of Neuromedicine and Movement Science, Faculty of Medicine and Health Sciences, Norwegian University of Science and Technology, Trondheim, Norway

ABSTRACT

Interoperable clinical decision support system (CDSS) rules are a pathway to achieving interoperability which is a well-recognized challenge in health information technology. Building an ontology facilitates the creation of interoperable CDSS rules, which can be achieved by identifying the keyphrases (KP) from the existing literature. However, KP identification for labeling the data requires human expertise, consensus, and contextual understanding. This paper aims to present a semi-supervised framework for the CDSS using minimal labeled data based on hierarchical attention over the documents fused with domain adaptation approaches. Then, evaluate the effectiveness of KP identification with this framework. In the view of semi-supervised learning, our methodology toward building this framework outperforms the prior neural architectures by learning with document-level context, no explicit hand-crafted features, knowledge transfer from pre-trained models (on unlabeled corpus), and post-fine-tuning with smaller gold standard-labeled data. To the best of our knowledge, this is the first functional framework for the CDSS sub-domain to identify the KP, which is trained on limited labeled data. It contributes to the general natural language processing (NLP) architectures in areas such as clinical NLP, where manual data labeling is challenging.

KEYWORDS

Clinical Decision Support System, Minimal labeled data, Hierarchical context, Semi-supervised learning, Transfer Learning, Domain adaptation

Abbreviations:

NLP: Natural language processing
CDSS: Clinical decision support system
HDE: Human domain expert
BiLSTM: Bidirectional long short-term memory
BiLM: Bidirectional language model
CRF: Conditional random field
GS: Gold standard
KP: Keyphrase

Code is available in GitHub: https://github.com/xjing16/cdss4pcp_nlpml_pipeline

Correspondence Author: Xia Jing, Email: xjing@clemson.edu

1. Introduction

Interoperability is a well-recognized barrier in health informatics. It creates chaos when transmitting patients' health records. Developing and maintaining such clinical decision support system (CDSS) rules across multiple healthcare settings is challenging. Having interoperable CDSS rules is one part of achieving interoperability in healthcare. An ontology using unambiguous concepts and their relationships can facilitate this process.

In a text article, these concepts are the orderly sequence of words or N-grams, namely, keyphrases (KP). These KP contribute not only to their meanings but also to the contextual understanding of the text. A KP is named a gold standard (GS) if it is selected by a human domain expert (HDE) after careful review and consensus among multiple HDEs.

An ontology can be constructed using such GS terms and their relationships, providing foundations for interoperable and generic CDSS rules [1]. However, an ontology construction is usually a manual process with the experts' input and curators' deep understanding of the domain and application contexts where KP identification is one of the steps. Automatic KP identification can be a critical complement to the aforementioned manual curation and construction of an ontology process.

We aim to build a system using natural language processing (NLP), which assists humans in identifying these KP faster. When the system design involves reviewing a text corpus and finding the data patterns to determine the N-grams, it can be driven by NLP neural network architectures [2], which can automate the identification of possible GS terms to match human intellect closely.

Given any text, some of the classic NLP algorithms (supervised and rule-based approaches) [3], require expensive labeled data to recognize the GS terms aligning with the HDE interests in CDSS concepts. Avoiding the need for labels, unsupervised algorithms [3] work with text similarity or semantic relatedness. With the growing corpus and increased contextual complexity towards text understanding, the prior approaches do not align with our interest in identifying the terms using contextual awareness.

Although Transformer models [4, 5] have been quite popular in accomplishing such a task using the context information with attention, they are computationally intense and require labeled data to fine-tune or perform a domain adaptation from biomedical to the CDSS domain. Leaving the computational complexity aside, the availability of high-quality human-labeled data is a problem. Only 1.2% of the data is labeled out of the total corpus from the CDSS literature. In the upcoming sections, we will discuss the effective usage of minimal labeled data to solve the challenges of domain adaptation which usually requires high-quality labels.

To avoid the above mentioned challenges, inferior neural architectures (compared to the Transformer [4,5] and other [6,7] models), such as long short-term memory (LSTM)-based encoders [8] along with the conditional random fields (CRF)-based decoder models [9] (a statistical modeling method for text pattern recognition, where current prediction is affected by neighbors), help identify the possible N-gram combination of tokens into a valid GS candidate term. It comes with the customization of numerous text features and attention levels over the text while recognizing the GS terms from the CDSS literature.

Focusing more on contextual understanding and bidirectional attention for LSTM enhances the prediction of KP [10]. However, adding document-level context during KP

identification leverages the broader contextual understanding of the text. Our approach is based on the NLP architectures presented by Zichao and Guohai et al. [11,12]. We used these NLP architectures as a base to create a hybrid approach by augmenting additional context layers for the existing attention-based BiLSTM-CRF [11,12] neural architecture to preserve its light-weighted heritage and append the benefits of context awareness. With this approach, we will see the details of harnessing the power of a minimal labeled dataset toward aligning the predictions with human interests. Furthermore, the main contributions of this paper are as follows:

- A hierarchical attention-based encoder (Hier-Attn-BiLSTM) neural network architecture, which incorporates the document-level information (with word-level and sentence-level contexts) in understanding the long-range contextual dependencies to identify KP.
- Creating high-quality synthetic labels to train the hierarchical attention-based model through the domain adaptation of a pre-trained Bidirectional Encoder Representations from Transformers (BERT) model.
- Understanding and harnessing the synthetic fine-tuning process when the machine learning (ML) model is limited by labeled data in a semi-supervised approach.
- Optimizing the process of using minimal human labels in KP identification based on experiments.

The remainder of this paper is structured as follows. We will discuss the related research in Section 2, formulate the task, describe the methodology and architecture in Section 3, outline the CDSS dataset, see the training procedure, illustrate the experiments and results in Section 4, depict the analysis and discuss the challenges identified during the implementation of this project in Section 5, and finally conclude in Section 6.

2. Background

2.1. CDSS Ontology

CDSS has been recognized and adopted broadly in healthcare settings due to its effectiveness in improving healthcare quality, adherence rates in medication prescription, and other clinical orders [13,14]. CDSS is usually a part of an electronic health record system. The CDSS rules are created by incorporating the clinical domain knowledge and contextual information, which affect the operational behavior of such decision systems' workflow. However, creating and maintaining these rules are tedious and challenging under resource-constrained conditions.

Creating an ontology facilitates the interoperability of CDSS rules and solves the challenges [1]. Traditionally, ontology construction is heavily expert-driven manual process. However, new terms and phrases emerge as the field and science evolve. Identifying such terms promptly can be a critical complementary component in building ontology, more comprehensively than merely manual curation. This work contributes to the basic workflow of ontology construction by automatically identifying KP within the process.

2.2. Similarity with Other NLP Problems

The NLP-based ML approaches deal with unstructured text data sources to extract structured information, understand the patterns, and identify the KP. Identifying a KP includes two phases: (1) extracting N-grams, limiting to noun phrases only, and (2) scoring or ranking the N-grams to find the best among selected ones to mark them as KP. Some of the popular methodologies for KP extraction, as given by Zhiyong He et al. [2], are summarized in the forthcoming sections, and we will discuss how the problem we focus on is different from theirs.

2.2.1. Statistical and Unsupervised Methods

In the case of limited labeled data, ML methods involving no training data — statistical or unsupervised, would be an ideal use case for us as proposed by Kazi et al. [3]. Some statistical features, such as TF-IDF [15, 16] and BM25 [17], describe the idea of differentiating the candidate terms into good or bad words. But they fail to deal with the unseen data distribution as the statistics are the conclusions drawn from the existing corpus.

In unsupervised methods, the KP are determined using semantic similarity, assuming that the more essential candidates cover all the important topics of the document. A graph is created using KP as nodes and their semantic similarity as the relations. Such a graph can be used by graph-based ranking algorithms such as Google's PageRank [18], MultiPartiteRank [19], PositionRank [20] and TopicRank [21] algorithms to retrieve the KP by scoring the terms across the relations drawn. However, the relation is given by the similarity measure between N-gram tokens, and it does not consider the document's contextual understanding to truly identify a KP.

2.2.2. Supervised Methods

Considering the KP identification as a classification task, we need training data to help the ML model align the predictions to the human interest. It can be coupled with carefully designed hand-crafted features to hold up to the expectations of improved term identification, to classify a term as either KP or non-KP, recasting it as a binary classifier. Furthermore, the popular choices among supervised algorithms are Decision Trees [23], Naïve Bayes [22], and Support Vector Machines (SVM) [24], which can be

used to solve binary classification. As the KP are not independent entities and are always an N-gram combination, they create chaos in the conceptual formulation of the problem.

Reformulating it as a ranking problem and marking top N entities as the KP, the research by Witten Ian et al. [25], a popular approach, KEA uses statistical features like TF-IDF and Word's First Occurrence Position (WFOP), whereas the work by Chengzhi Zhang et al. [26] exploded into the usage of features such as TF-IDF and WFOP with length of token and linguistic features such as Part of Speech (POS) [27] tags to normalize the position and occurrence of the KP. A linear ranking SVM was used to rank the KP [28]. The BiLSTM-CRF model [29] considers it a sequence tagging problem and extracts the KP with superior performance [30]. However, direct implementation of supervised methods does not solve our problem, as labeled data limits us, and we will further discuss this in the upcoming sections.

2.2.3. Named Entity Recognition (NER)

Any sub-task of information extraction and retrieval classifying the N-gram entities into pre-defined categories, such as drug, gene, disease, ORG, person, location, and data, is known as NER [31]. It can be broken down into two problems, entity identification and entity classification, similar to KP extraction. The first identification phase is N-gram segmentation, where N-gram can be the sequence of tokens. The second classification phase is similar to organizing the respective terms into categories. Here, all the entities can be distinct and exist independently.

While the problem statement looks similar, our task of identifying the top KP for the document has no pre-defined categories like NER. Although it is based on a contextual understanding of the text and often comprehended by confidence toward classifying an entity into one of the pre-defined categories, the entities do not reveal their significance in document understanding. Additionally, in the KP, we need to include not only nouns but also verbs. Therefore, our task is not the same as traditional NER but is close to entity recognition.

2.3. Domain Adaptation

Entity recognition or KP identification can be a fundamental task for various NLP applications, such as entity identification, building an ontology, and knowledge graph construction over entities. Using a large amount of labeled data to train the ML model from scratch is challenging. To avoid this problem, domain adaptation [32,33] helps fine-tune the pre-trained ML models in the parent or similar domain to the sub-domain citing the minimal usage of labeled data as a standard practice.

A popular model in entity identification, Spacy [34], based on BERT [5], is trained on OntoNotes5 [35] and WordNet [36] corpora. Although it works very well with language modeling and text understanding for English corpora, it fails to identify the entities in sub-domains (e.g., biomedical, and clinical informatics). The large language model (LM) is domain-adapted with 785 K vocabulary and 600-word vectors into sciSpacy, specializing in identification of biomedical entities [38]. We use this sciSpacy model to further fine-tune it to the CDSS sub-domain and strengthen the transfer learning approaches in our methodology.

2.4. Language Model (LM)

The LM is one of the critical aspects of present-day NLP architectures [39] which is based on the hidden Markov chain models (HMM) [40] and used where the labeled training data is limited. It is a statistical and probabilistic technique to determine the conditional probability of each word's occurrence in a given sentence. To create such an LM, all the sentences in the document are unified into one by removing punctuation. Then, we slide over the word windows to train an LM without needing labeled data to understand the context of the words and their characteristics. Here, we need to understand domain-specific language and the distribution of words, a CDSS domain in our case. We can use this trained LM to transfer its neural network parameters to the actual model, helping it learn the language distribution for the CDSS domain [41].

3. Methodology

Our task is similar to sequence labeling, and with the growing popularity of neural networks, hand-crafted text features (TF-IDF, Length of token, POS tags, WFOF etc.,) are no longer needed. A bidirectional long short-term memory (BiLSTM) over a sequence of words with textual features as the encoder and a CRF layer as the decoder can learn the N-gram entity patterns and their occurrence with context over the current sentence [11,12].

Of all the textual features, word embeddings (WE) play a significant role in transforming text information into mathematical representation to feed input data into deep learning models. We propose a hierarchical attention-driven context added to each word to improve the inference and learn a variety of text patterns with minimum labels to bridge the gap of contextual understanding for word representations. Details of the newly proposed methodology are presented in the following sections.

3.1. Overview

3.1.1. Defining the Task

KP identification is a typical sequence labeling task where we find the N-gram KP from the document. Usually, a document with m sentences, $d = (s_1, s_2, \dots, s_m)$ and each sentence containing n tokens or words, $s_i = (w_{i1}, w_{i2}, \dots, w_{in})$ is the input to the model and output $z_i = (z_{i1}, z_{i2}, \dots, z_{in})$ would be a sequence of tags in BIO token tagging [42] representation.

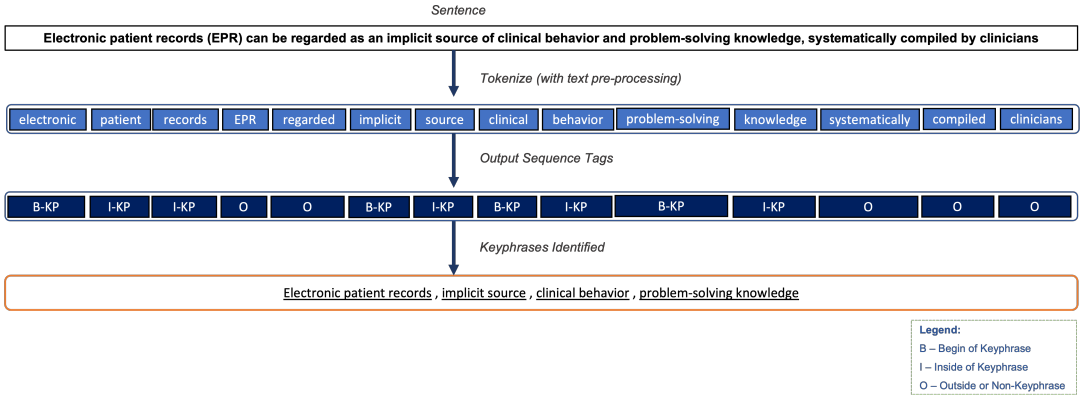

Figure 1. Flow of labeling Keyphrases (KP) from a sentence

In BIO token tagging [42], the first word is labeled B-KP, the remaining words in an N-gram phrase are labeled I-KP, and the rest of the non-KP tokens are marked as O. For example, as shown in Fig. 1, the input document (electronic, patient, records, epr, regarded, implicit, source, clinical, behavior, problem solving, knowledge, systematically, compiled, clinicians). The model can output the sequence tag (B-KP, I-KP, I-KP, O, O, B-KP, I-KP, B-KP, I-KP, B-KP, I-KP, O, O, O) where the keyphrases can be generated by decoding the outputs tags (electronic patient records, implicit source, clinical behavior, problem-solving knowledge).

Here, we leverage the document-level context by combining hierarchical attention (i.e., adding word-level and sentence-level attentions in a hierarchical fashion to create the document vector), improving the performance. Thus, all the sentences in the form of embeddings followed by their corresponding attentions will be used to complement the understanding of the current sentence. In simple words, the input to our model will be all the sentences from a single document, and for each sentence, we find its relevance compared to other sentences and their words from the within the same document to calculate hierarchical attention in understanding the context.

3.1.2. High-Level Design

Our approach to ML model architecture includes creating synthetic training data, pre-training, encoder with neural attention + decoder, and fine-tuning with actual labeled data as illustrated in Fig. 2. Inspired by Guohai et al. [12] and Saad et al. [43] research works, firstly, we train the WE model and bidirectional language model (BiLM) as shown in Fig. 3 using unlabeled data and then transfer their knowledge into the actual model's initial layers for embedding and LSTM respectively. Secondly, all the sentences from the single document are fed into the model as a batch at a given time, where each word in the sentence is transformed into a vector with the WE model. Then, we introduce the abstraction of hierarchical attention — attention at various levels, namely at the word and sentence levels — to aggregate them into sentence and document-vectors, respectively.

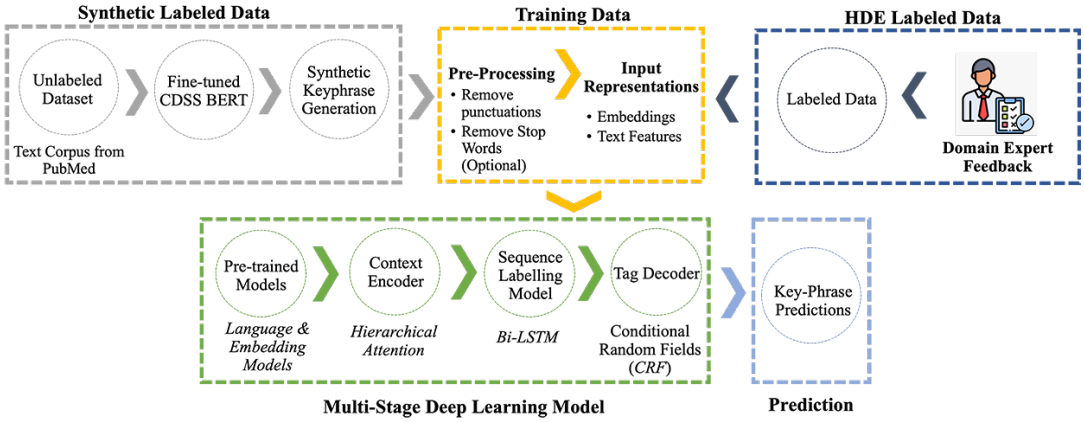

Figure 2. High-Level Design of the methodology.

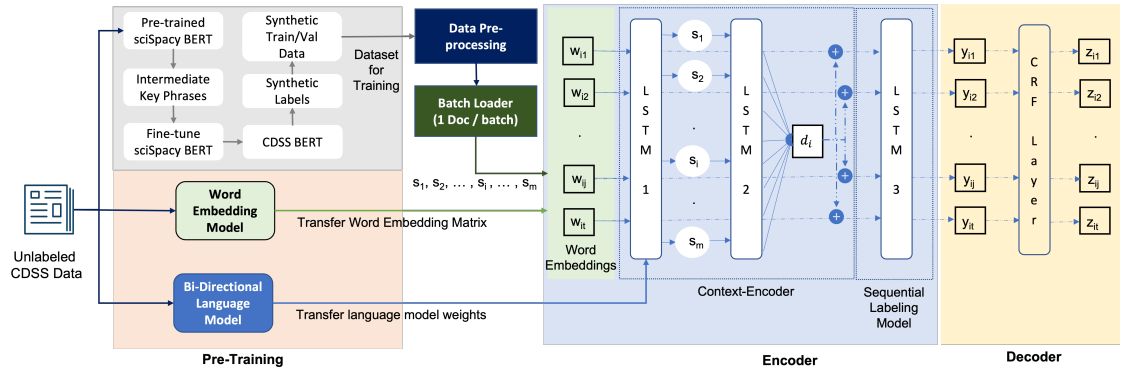

Figure 3. Multi-stage deep learning model based on hierarchical context and transfer learning

Using these embedding and attention vectors, we calculate the hierarchical attention for any given word using the second LSTM, which is further deduced into the final LSTM network along with the outputs of first LSTM (BiLM) network completely, encoding the one document-at-a-time. Lastly, we use the encoded information to feed the Conditional Random Fields (CRF) layer which then decodes the best probable sequence decisions to mark the output labels with BIO token tag representations which are used to group the tokens and identify the KP. Later, we turn to fine-tune the model to enhance its performance using a minimally labeled dataset.

3.2. Synthetic Data

While we lack a labeled dataset, domain-adapted or fine-tuned models can be used to create labels i.e., synthetic labels which helps us to train the initial ML model. Later, the model can be fine-tuned with the HDE labels to identify the actual keyphrase, avoiding the potential exhaust of HDE labels during initial rounds of training.

To achieve this, firstly, we perform *domain adaptation* of a sciSpacy BERT model [38] by generating the KP (intermediate) on the CDSS dataset and use them to adapt the sciSpacy BERT to the CDSS sub-domain. Then, we generate the **KP (synthetic)** on the CDSS dataset and mark the labels in the BIO format on the texts without HDE labels, namely the synthetic dataset with labels for the CDSS sub-domain. We use this dataset to train and test our ML model, as discussed in the following sections.

3.3. Pre-Training

3.3.1. Word Embedding (WE) Model

A WE is mathematical vector representation of a given word, which ensures minimal distance between the vectors with words of similar meaning. These embeddings capture the language semantics and syntactic information using the Word2Vec [44] *skip-gram* approach and are used as input to train deep learning models. We have also experimented with creating fastText [45, 46] and GloVe [47] embeddings as alternative embedding models to evaluate the performance difference between them in our approach.

3.3.2. *Bi-Directional Language Modelling (BiLM)*

To learn the probability distribution over sequences of words, we use a shallow layered bidirectional recurrent neural network [48] (e.g., LSTM and GRU) to learn the joint probabilities represented by WE. To ensure that the network learns such a distribution, we measure its perplexity. Such a network that learns the word distribution is known as the BiLM [39]. It computes the conditional probability of occurrence of the next word(w_i) based on the previous($w_1, \dots, w_{i-1}$) and future words($w_{i+1}, \dots, w_n$) in a sentence(s) as shown in Eq. (A. 1),(A. 2) [39], where each sentence(s) is represented by last word's context (given by LSTM's cell state) in both left($\overleftarrow{c}_n^{LM}$) and right($\overrightarrow{c}_n^{LM}$) directions. Here Eq. (A. 2) is the probability of LM in the reversal order when compared with the Eq. (A. 1).

$$p(w_1, w_2, \dots, w_n) = p(w_2|w_1) \dots p(w_n|w_{n-1}) = \prod_{i=2}^{n} p(w_i|w_1, w_2, \dots, w_{i-1}) \quad (A.\ 1)$$

$$p(w_n, w_{n-1}, \dots, w_1) = p(w_{n-1}|w_n) \dots p(w_1|w_2) = \prod_{i=n-1}^{1} p(w_i|w_n, w_{n-1}, \dots, w_{i+1}) \quad (A.\ 2)$$

$$s = [\overleftarrow{c}_n^{LM}; \overrightarrow{c}_1^{LM}] \quad (A.\ 3)$$

Both the forward and backward LSTM encode the history of previous tokens in the respective directions into fixed dimensional vectors ($\overleftarrow{h}_{i-1}^{LM}$, $\overrightarrow{h}_{i-1}^{LM}$) for a given word($w_i$), where a soft-max layer maximizes the likelihood(p) of the word(w_i) in the given sentence(s) in the corpus. After training, a BiLM can represent the sentence of a document by concatenating the last cell (i.e., the last word of the sentence) state carrying the context in either direction to represent the input sentence as shown in Eq. (A. 3).

3.4. *Hierarchical-Attention-BiLSTM-CRF Model*

3.4.1. *Encoder*

As illustrated in Fig. 4, this architecture is adopted from those presented by Zichao Yang, Guohai Xu and Luo L et al. [11, 12, 49]. In contrast to the prior works, we are limited by the labeled data. To adeptly use the labels, we encode one document at-a-time to capture document-level context with a stacked BiLSTM [11]. Here, the rudimentary layers of stacked BiLSTM are initiated with a transfer strategy from pre-trained WE and BiLM models' weights.

The embedding and first LSTM layers in our encoder share the same architecture as the pre-trained models, which can seamlessly *transfer the model parameters or weights* between the models [12]. Using the transfer strategy, our model can efficiently initialize and learn from the synthetic dataset in identifying the KP.

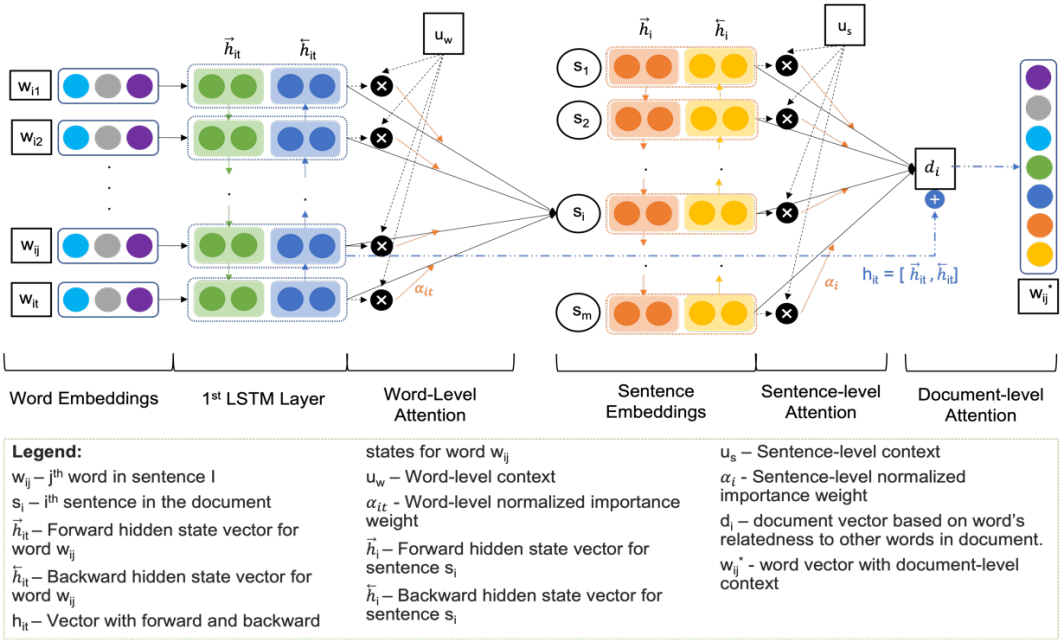

Figure 4. Word encoding with Hierarchical-Attention-BiLSTM with Document-level context

We use all the sentences of one document, $d = (s_1, s_2, \dots, s_m)$, each sentence $s_i = (w_{i1}, w_{i2}, \dots, w_{in})$ and each word $w_{it} \forall t \in [1, n]$. We embed the words into a vector (x_{it}) through an *embedding matrix* (W_e). BiLSTM summarizes the bidirectional context information as shown in Eq. (B. 1)(B. 2)(B. 3) where each word's hidden state (h_{it}) is obtained by concatenating the forward $(\overleftarrow{h}_{it})$ and backward $(\overrightarrow{h}_{it})$ hidden state vectors, i.e., $h_{it} = [\overleftarrow{h}_{it}; \overrightarrow{h}_{it}]$. Here the hidden state vector provides sentence-level context to each word [12].

$$x_{it} = W_e . w_{it} \forall t \in [1, n] \quad (B.\ 1)$$

$$\overrightarrow{h}_{it} = \overrightarrow{LSTM}(x_{it}) \forall t \in [1, n] \quad (B.\ 2)$$

$$\overleftarrow{h}_{it} = \overleftarrow{LSTM}(x_{it}) \forall t \in [1, n] \quad (B.\ 3)$$

We calculate word similarity (u_{it}) using a neural network's parameter for weighted matrix (W_w) and word representation (h_{it}) given by BiLSTM along with bias (b_w) [50]. Then we calculate the **word-level attention** by aggregating the h_{it} and u_{it}) using a word-level context vector (u_w) to get a word-level normalized importance weight (α_{it}). Finally, we compute the sentence vector (s_i) as a weighted sum of word representations as shown in Eq. (C. 1)(C. 2)(C. 3). Initially, u_w is the neural network parameter with random initialization and learned during the training process.

$$u_{it} = \tanh(W_w . h_{it} + b_w) \forall t \in [1, n] \quad (C.\ 1)$$

$$\alpha_{it} = softmax(u_{it}^T . u_w) = \frac{\exp(u_{it}^T . u_w)}{\sum_t \exp(u_{it}^T . u_w)} \forall t \in [1, n] \quad (C.\ 2)$$

$$s_i = \sum_t \alpha_{it} . h_{it} \forall t \in [1, n] \quad (C.\ 3)$$

Similarly, a document vector can be computed using **sentence-level attention** over the sentence vectors (s_i) [11,12] using a second BiLSTM network and thereby concatenating the forward ($\overleftarrow{h}_i$) and backward ($\overrightarrow{h}_i$) states to encode a sentence, $h_i = [\overleftarrow{h}_i; \overrightarrow{h}_i]$ based on neighbor sentences as shown in Eq. (D. 1)(D. 2).

$$\overrightarrow{h}_i = \overrightarrow{LSTM}(s_i) \forall i \in [1, m] \quad (D.\ 1)$$

$$\overleftarrow{h}_i = \overleftarrow{LSTM}(s_i) \forall i \in [1, m] \quad (D.\ 2)$$

As shown in Eq. (E. 1)(E. 2)(E. 3), to estimate the sentence-level context vector (u_s), firstly we use neural network parameter for weighted matrix (W_s), sentence representation (h_i) and bias (b_s) to calculate sentence-similarity (u_i). Secondly, we randomly initialize u_s and learn it during training, to calculate the sentence-level normalized importance weight (α_i), which yields a **document vector(d_i)** for each word representing which sentences are important for a given word to consider while identifying it as a KP as provided [11].

$$u_i = \tanh(W_s . h_i + b_s) \forall i \in [1, m] \quad (E.\ 1)$$

$$\alpha_i = softmax(u_i^T . u_s) = \frac{\exp(u_i^T . u_s)}{\sum_i \exp(u_i^T . u_s)} \forall i \in [1, m] \quad (E.\ 2)$$

$$d_i = \alpha_i . h_i \forall i \in [1, m] \quad (E.\ 3)$$

Unlike the previous work as proposed by Guohai Xu et al. [12], finally we concatenate the first LSTM's hidden local state(h_{it}) with the document vector (d_i) into a new vector $[h_{it}; d_i] \forall t \in [1, n]$, based on given the word's relatedness to other words in the document, in specific providing document-level context to each word. Next, the extended representation will be further used by the final LSTM layer to identify the labels.

3.4.2. Decoder

As described by Ling Luo et al. [49], we use the CRF [9] layer as the decoder to produce the confidence scores for the words having each possible label as the output score of the decoder. Given the transition and network scores, we make tagging decisions independently, considering P the matrix of scores of the network output.

The score of sentence (s_i), along with a sequence of predictions $y_i = (y_{i1}, \dots, y_{ij}, \dots, y_{it})$, is given by the sum of transition scores and network scores as shown in Eq.(F. 1). Here each $P_{a,b}$ represents the matrix of scores of b^{th} tag of the a^{th} word in the sentence. Furthermore, tagging transformation matrix (T) is trained as the model parameter. Here $T_{a,b}$ represents the transition score from tag a to tag b through successive words where $T_{0,b}$ is the initial score for the starting from tag b.

To yield the conditional probability of the path y, we normalize the score for all possible paths using a soft-max function using Eq. (F. 2). Then, we maximize the log probability of valid tag sequences. We can obtain the maximum score using the dynamic programming approach of Viterbi decoding [51] for the best tag path given by Eq. (F. 3).

$$score(s_i, y_i) = \sum_{k=0}^{n} \left(T_{y(i,k-1),y(i,k)} + P_{(k,y(i,k))} \right) \quad \forall i \in [m, 1] \tag{F. 1}$$

$$p(y_i|s_i) = \frac{\exp\left(score(s_i, y_i)\right)}{\sum_{\tilde{y}_i} score(s_i, y_i)} \quad \forall i \in [m, 1] \tag{F. 2}$$

$$z_i = \underset{\tilde{y}_i}{argmax}\left(score(s_i, y_i)\right) \ \forall i \in [m, 1] \tag{F. 3}$$

3.5. Fine-Tuning with Gold Standards

Now with the newly modified neural architecture capable of learning from document-level understanding provided by hierarchical context at both word-level and sentence-level attentions, we used this further to train it with synthetic data first and fine-tune it later with GS Annotations to align the model's predictions to HDE expectations. The new architecture helps the model discriminate the input data by document-level context when identifying an entity and optimize the GS Annotations required to fine-tune. We will discuss the effectiveness of the new methodology through results in the upcoming sections.

4. Experiments and Results

4.1. Dataset

The text corpus was obtained from PubMed by filtering the CDSS literature in the MEDLINE format. Of these research articles, those with a valid PMID (PubMed Identifier for a unique article) were retained for XML parsing, where we retained the MeSH (Medical Subject Headings) terms and associated them with corresponding articles. Tables 1 and 2 show the details of the dataset. Appendix A shows detailed information on the dataset at various levels of text pre-processing phases.

Table 1. Details of the CDSS Dataset.

-	Full CDSS (FC)	FC with PMIDs	GS (+8 ACM)	No/Little Abs.	Final/Total Dataset
Articles	3545	3326	133	99	3281

No/Little Abs. - Abstracts having less than 3 sentences including the title

Table 2. CDSS Dataset: train, validate and test.

-	Total	Unlabeled (with Synthetic KP)	Train	Validate	Test (GS91)	GS(GS42)
Articles	3281	3148	1049	2099	91	42

Here GS91 and GS42 are 2 sets of HDE-labeled datasets.

In the total dataset (3545), after parsing the information from PubMed in XML format, we were left with 3326 articles containing duplicates of the GS dataset. Furthermore, we had 2 sets of human-labeled datasets (GS91 and GS42). We used GS91 as unseen data to compare the model's performance, and the GS42 was used to fine-tune the ML model later. During the pre-processing of the articles, we removed the abstracts with little or no abstracts by checking the minimum number of sentences (< 3). Now, only 3148 unlabeled articles remained, and we created synthetic KP and marked the labels to create a synthetic labeled dataset for the CDSS domain with a 1:2 train-validation split. Cohen's kappa rates for the first 42 (GS42) abstracts were 0.93 (between annotators 1 and 2) and 0.73 (between annotators 1 and 3) [37]. For the second set of abstracts (GS91), Cohen's kappa rates were 0.87 (between annotators 1 and 2) and 0.97 (between annotators 1 and 3).

4.2. Synthetic Label Creation

To ensure the high-quality creation of synthetic labels, we experimented with different unsupervised algorithms (namely, PositionRank, MultiPartiteRank, and TopicRank) and NER (i.e., sciSpacy) to identify the KP from a given text and compared them with the manual labels to compare the performance. The results are shown in Table 3. We found that BERT-based sciSpacy [38] outperforms other unsupervised methods in generating the KP close to HDE labels.

Table 3. Evaluation of generated synthetic KP with different approaches

Approach	Accuracy	Misclassification	Precision	Recall	Specificity	F1-Score
sciSpacy	0.69	0.31	0.36	**0.81**	0.66	**0.50**
PositionRank	0.76	0.23	**0.39**	0.36	0.86	0.38
MultiPartiteRank	0.76	0.24	0.38	0.36	0.86	0.37
TopicRank	0.77	0.23	**0.39**	0.36	**0.87**	0.37

4.3. Preparation

4.3.1. CDSS Domain Adaptation

We made domain adaptation for the sciSpacy LM (from biomedical entities to CDSS entities) to make it more suitable for the CDSS context [32,33]. To fine-tune the model, we used the synthetic labels created from the sciSpacy model's prediction on CDSS unlabeled corpus, like a semi-supervised approach as proposed by Syed et al. [52]. To unfold the challenges of creating a better-fine-tuned model, we fine-tuned it at different levels and different dataset combinations to look for the optimal one (Table 4 and Fig. 5). Of the experiments conducted, we found that Level 1 fine-tuning of the sciSpacy model with the synthetic dataset yielded better results, and further fine-tuning overfitted the model's predictions.

Table 4. Evaluation of fine-tuning sciSpacy model for CDSS

Fine-Tune	Base	Model	Train Dataset	GS Dataset	Precision	Recall	Accuracy	F1-Score
Level 0	sciSpacy	sciSpacy (en_core_sci_lg)	PubMed	42	0.61	0.18	0.93	0.27
				91	0.59	0.23	0.97	0.33
				133	0.62	0.22	0.96	0.33
Level 1	sciSpacy	cdssSciSpacy	Synthetic CDSS (1866 Train / 622 Val)	42	0.70	0.38	0.97	0.5
				91	0.73	**0.64**	**0.99**	**0.68**
				133	**0.74**	0.59	**0.99**	0.66
Level 2	cdssSciSpacy	cdssSciSpacy GS42	42 GS (33 Train / 9 Val)	91	0.57	**0.64**	**0.99**	0.60
Level 2	cdssSciSpacy	cdssSciSpacy GS91	91 GS (72 Train / 19 Val)	42	0.66	0.38	0.97	0.48
Level 2	sciSpacy	sciSpacy GS42	42 GS (33 Train / 9 Val)	91	0.57	0.54	**0.99**	0.55
Level 2	cdssSciSpacy	cdssSciSpacy GS66 [1]	133 GS (52 Train / 14 Val)	67	0.63	0.62	**0.99**	0.62

[1]Repeated experiment 50 Times on random samples of GS 133.

4.3.2. Token Tagging Representation

To identify an N-gram sequence, we took the help of token tagging systems where each KP is marked with either BIO or BILOU encoding schema [42] to represent the text chunks effectively. Here, we conducted the experiments on both schemas to identify the better one that fits the CDSS corpus, and the results are shown in the Table 5. As both the token tagging representations have near-similar performance metrics, we chose to stay with BIO token tagging for the label marking because, for the CDSS domain it outperforms BILOU by a slight margin in F1-Scores.

Table 5. Entity-level metric evaluation - token tagging

Encoding Schema	Dataset	Precision	Recall	Accuracy	F1-Score
BIO	Validation Dataset (Synthetic) Labels	0.75	**0.68**	**0.92**	**0.71**
	GS42 Labels	0.60	**0.50**	**0.88**	**0.54**
	GS91 Labels	0.61	**0.50**	**0.88**	**0.55**
BILOU	Validation Dataset (Synthetic) Labels	**0.76**	0.60	0.92	0.69
	GS42 Labels	**0.60**	0.41	0.87	0.49
	GS91 Labels	**0.65**	0.42	0.86	0.51

4.3.3. Stemming vs. Non-Stemming

Stemming the vocabulary was one of the normalization techniques involved in pre-processing the text data before we feed it to ML models. It represents the morphological structure of the language. For the English corpora, while stemming operation seems to benefit the document indexing, sometimes it can worsen the effects of the topic understanding [53]. To analyze the effect of stemming on CDSS corpora, we've experimented with the performance of KP identification concerning stemmed and non-stemmed KP on both the synthetic labels and GS-labeled data. The results are shown in the Tables 6, 7. The results indicated that the performance of the ML models deteriorated with the stemming of words, so we opted for non-stemming in the text pre-processing steps.

Table 6. Comparison of stemming evaluation on Validation Dataset (Synthetic).

Metrics	Validation Data Labels					
	Non-Stemming			Stemming		
	B-KP	I-KP	O	B-KP	I-KP	O
Accuracy	0.87	0.92	0.92	0.85	0.91	0.90
Misclassification	0.13	0.09	0.08	0.16	0.09	0.10
Precision	0.85	0.80	0.91	0.83	0.74	0.87
Recall	0.92	0.76	0.81	0.92	0.55	0.77
Specificity	0.83	0.96	0.96	0.74	0.97	0.95
F1-Score	**0.88**	**0.78**	**0.86**	0.87	0.63	0.82

Table 7. Comparison of stemming evaluation on GS42 Dataset.

Metrics	GS42 Labels					
	Non-Stemming			Stemming		
	B-KP	I-KP	O	B-KP	I-KP	O
Accuracy	0.57	0.86	0.51	0.52	0.86	0.53
Misclassification	0.44	0.15	0.49	0.48	0.14	0.47
Precision	0.30	0.40	0.83	0.32	0.41	0.81
Recall	0.74	0.67	0.35	0.80	0.36	0.33
Specificity	0.52	0.88	0.85	0.42	0.93	0.47
F1-Score	0.43	**0.50**	**0.49**	**0.45**	0.38	0.47

4.3.4. Loading Pre-Trained Models

As discussed in the Section 3.4.1 and shown in Figures 2 and 3, we first trained the WE model and the BiLM separately on the unlabeled corpus and then, using the approach of transfer learning, updated the weights of the encoder's initial layers before we started training the model. After training, we exported the parameter weights of both models individually and imported them later into the LSTM network.

4.3.5. Training

After obtaining synthetic labels generated from the best performing domain-adapted model (as mentioned in the Sections 4.2 and 4.3.1), we labeled the KP with the BIO format [42] to start the ML model training procedure for 30 epochs. Then, we evaluated the sequence-level entity metrics using standard ML metrics, i.e., Precision, Recall/Sensitivity, F1-Score and Accuracy. The parameters and configurations of the Hier-Attn-BiLSTM-CRF neural network model are as follows:

- WE Dimension: 300
- LSTM hidden layer dimension: 256
- Dropout Ratio: 0.2
- Epoch: 30 (number of times every document is shown to ML model)
- Batch Size: 1 (one document at a time shown to model, to calculate the context with documents having varying sentences with 52 sentences being the maximum for single abstract)
- Max sentence length: 128 (For CDSS corpus, maximum words per sentence are 105)

- WE Type: Word2Vec
- Text pre-processing: remove stop words and punctuation
- Stemming: no
- Train-validation split: 1:2
- Pre-trained sciSpacy BERT model: en_core_sci_lg

4.4. Evaluation

4.4.1. Leveraging Document-level Context

Understanding the textual context while KP identification was significant in predicting relevant candidate terms from a given text [10, 54]. To understand that, we've experimented with the use of different encoding combinations for word-level, character-level embeddings and CNN-based text features (Length, POS tag, Text Rank, TF-IDF Score and Position of First Occurrence [55]) compared with the hierarchical-attention and sentence-level embedding working at the document-level context. The results are shown in the Table 8 and Fig. 5. The experiment showed that our methodology, which included a hierarchical context-driven model, had improved metrics over the base BiLSTM-CRF model and gave a head-to-head competition to the other models with Character Embedding and CNN-based Text Features by only losing with the lesser Recall values.

Table 8. Comparison of evaluations on different contextual level attention

Model	Encoder Details	Experiment Runs	Train Dataset	Test Dataset	Precision	Recall	Accuracy	F1-Score
BiLSTM-CRF	BiLSTM(Word Embd's)	1	1049 Synthetic	2099 Synthetic	0.72	0.66	**0.92**	0.69
				42 GS	0.54	0.46	0.86	0.49
				91 GS	0.59	0.48	**0.88**	0.53
BiLSTM-CRF	BiLSTM(Word Embd's) + BiLSTM(Char Embd's)	1	1049 Synthetic	2099 Synthetic	0.70	0.70	0.85	0.70
				42 GS	0.52	**0.56**	0.78	0.54
				91 GS	0.58	0.53	0.77	**0.55**
BiLSTM-CRF	BiLSTM(Word Embd's) + BiLSTM(Char Embd's) + CNN(Text Features)	1	1049 Synthetic	2099 Synthetic	0.73	**0.71**	0.85	**0.72**
				42 GS	0.56	0.55	0.78	0.55
				91 GS	0.58	**0.55**	0.78	0.57
Hier-Attn-BiLSTM-CRF	BiLSTM(Word Embd's) + Hierarchical Context (word-level sentence-level attentions)	1	1049 Synthetic	2099 Synthetic	**0.75**	0.68	**0.92**	**0.71**
				42 GS	**0.6**	0.5	**0.88**	0.54
				91 GS	**0.61**	0.5	**0.88**	**0.55**

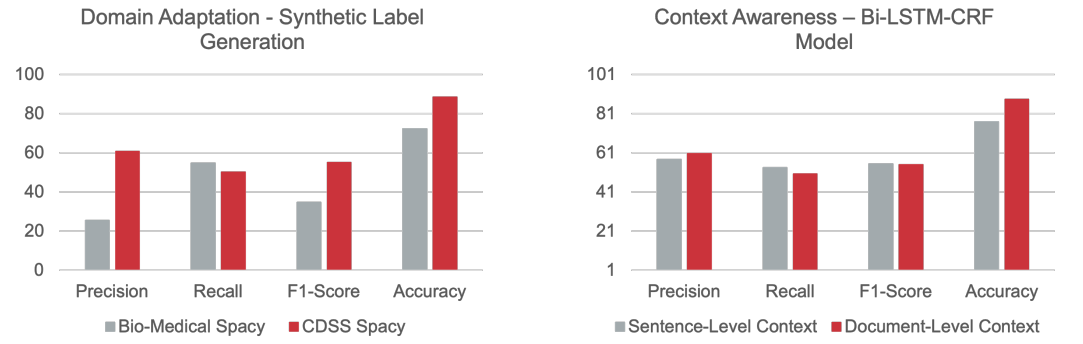

Figure 5. Comparison of results for domain adaptation and hierarchical context (document context through word-level and sentence-level attention).

4.4.2. Fine-tuning with Gold Standard (GS) Labels

We harnessed the semi-supervised learning approach and further fine-tuned the Hierarchical-Attention based BiLSTM-CRF model to strengthen its prediction [52]. The experiment included adding the truth to synthetic data proportions at intervals with multiples of 2 between 2 to 8 documents with true or GS labels are shown for every batch of 100 synthetic labeled documents. It measured the performance of learning with human feedback over the iterations of ML model training by running independent experiments for 10 and 50 times. As shown in Fig. 6, the results demonstrated that exposing 2-4 true samples to 100 synthetic samples enabled the model to learn more efficiently from the minimum labeled dataset. The tabular details of the performance metrics for the pictorial representation are shown in Appendices B and C.

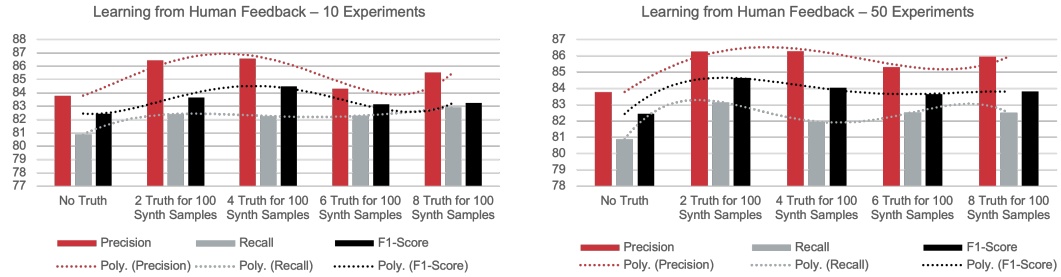

Figure 6. Results for fine-tuning with Gold Standard (GS) Labels

5. Discussion

While the traditional decision support systems have been designed as rule-based or semantic-driven applications, identifying such rules to match human expertise is complex due to the collaborative consensus between the labels curated. To create a rule, finding such patterns in the text is tedious for humans and leaves out a scope for us to automate. Regarding processing texts efficiently and discovering patterns, ML algorithms have unparalleled advantages and an ability to learn from minimal human labels, as they have brighter and broader potential in many application areas.

Identifying the KP that have higher significance in summarizing text is a different task from ours. Even though it is similar on the surface, it is completely different underneath because the KP are not a fixed set of terms that can be objectively marked as correct. Despite their importance in text understanding spanning the best coverage and relevance to the given text, manual labeling is expensive, and automating human understanding of the text is challenging, especially in highly specialized fields, such as medicine.

Although various ML approaches have evolved to solve some of the common problems of the text, they fail to understand the context behind the annotation due to the long-range dependencies of the natural language. To solve this problem, we used a semi-supervised approach with hierarchical attention over text to provide a larger but still focused context (one document) to the model while working with a word. As we advance, we will discuss the contributions, interpret the experiment results, and challenges of this work.

5.1. Result Interpretation

The lack of labeled data is one of the key challenges of this project. Thus, we used semi-supervised learning approaches to synthetically create the labels from a pre-trained model and compare them with a domain-level fine-tuned model. To assess the quality of the synthetic labels, we experimented with different approaches, including unsupervised ranking approaches [18–21] and the pre-trained models based on the Transformer neural architectures [4, 5]. We found that sciSpacy [38] (BERT model) outperforms the others in matching the synthetic labels to the GS candidate terms, as shown in Table 3. Although the F1-Score of 0.5 does not seem ideal, we must recognize that this is a very challenging task, even for HDE. Although the task can be presented as a simple yes or no task, the actual identification is far more complicated than a binary task. The HDE has to use rich background knowledge to make the judgment.

To strengthen the pre-trained model's performance in generating the synthetic labels, we adapted it to the CDSS domain. We observed that fine-tuning with synthetic labels provided us with much better labels than the naive sciSpacy model. To examine it more closely, we performed supplementary fine-tuning with different combinations (42 GS, 91 GS and 67 GS - different train-validate-test data subsets each time from 133 GS) of the labeled dataset where each combination is trained freshly at the respective fine-tuning level (Table 4). We understood that incremental fine-tuning introduces variance into the LM and increases perplexity to drop its performance further as the fine-tuning levels grow.

To efficiently represent the token to the ML model, we experimented with encoding schema's - BIO and BILOU token tagging representations to identify that there are no significant differences in the performance for the CDSS corpus, but BIO slightly performs better (Table 5). Although standard approaches in the NLP pre-processing include either stemming or lemmatization resulting in high performance, in our case, it deteriorates the topical understanding and inference of the ML model. We experimented with the ML model's performance on stemmed vs. non-stemmed tokens, and the results shown in Tables 6 and 7 align with our conceptual understanding.

Once the words are tokenized, we need embeddings to bind the token information to a vector to feed it further to the model. Most WE models work with vocabulary from the existing text corpus and fail to handle Out of Vocabulary (OOV). To solve the OOV problem, we can use sub-word information with character N-grams using fastText. Using fastText reduces the length of vocabulary as it remembers sub-word information. While the total vocabulary with Word2Vec is around 15.8 K, fastText has only 4.7 K sub-words. Also, it only shows a 0.5%-1% improvement, as given by Benedict et al. [56]. Therefore, we reverted to the older Word2Vec approach for pre-training the WE model as it is easier to transfer the embedding matrix weights between pre-trained and actual models. Our methodology uses index-to-token and token-to-index mapping while encoding the words, and the length of the vocabulary(L) is further used as square matrix dimensions of the WE (W_e) matrix, and it helps us find the similarity between any two words.

We now introduce the word-level attention mechanism because not all words contribute equally to the meaning of the sentence. Then, we aggregate the representation of those words to form a sentence vector, which is in return used to create a document vector for each word in the broader context of the document and its sentences. We conducted experiments on the different input encoders (word-level, character-level, and text-based CNN) with the hierarchical attention-based model and a

CRF decoder. Furthermore, we evaluated the performance of the neural network with input word representations bearing the document-level context to understand long-range dependencies of the text and the results are shown in Table 8 and Fig. 5. Although the metrics are on-par with the other models, our model had no hand-crafted features except the pre-training for WE and LM.

With the increase in the feature representations, the complexity of the evaluation will be tricky and no longer work at the token level. Still, it needs to be evaluated on the entity level with a complete match of the GS label as proposed by Nancy et al. [57]. Therefore, we used the sequence labeling evaluation given by Hiroki et al. [58] to decipher the results. As shown in Fig. 4, our model with hierarchical attention improved Accuracy by 10% compared to the base model without any character-level and textual features, suggesting an improvement in the performance due to the added hierarchical context provided to each word representation. We also noted that it has an improved Precision over the remaining models and allowed us to maintain the F1-Score even with the decline in Recall values.

To further strengthen the ML model, we used fine-tuning with the GS labels to align the model's predictions from synthetic to GS labeling. To evaluate the model's performance after training, we kept aside 91 GS and only fine-tuned the model with 42 GS labeled documents from the CDSS corpus by varying the true labeled documents shown (0/2/4/6/8 GS) for every 100 labeled documents during the model training process, marking the essence of minimal true labels shown. As shown in Fig. 6, a poly-fit curve over the scores concludes that showing 2-4 true samples for 100 synthetic samples during ML model training demonstrates better performance. The experiments and the results guide us to optimize our model and settings for the operation, and we hope the results can be a reference point for others to plan their NLP tasks.

5.2. Challenges

The generation of manually labeled data is not only expensive but also is a labor-intensive and demanding task in the fields like medicine. Circumventing this issue, a small set of samples can be labeled by humans. To create such a small set of high quality labeled data, picking the samples from different concepts of the CDSS sub-domain is significant because it helps the model to efficiently learn from the diversified samples and their labels provided. Therefore, effectively selecting the data for human annotation exposes us to one of the ubiquitous problems of the selection bias, over picking the articles from the CDSS corpora. Also, we encounter the same problem in the selection of data samples or documents for the fine-tuning process.

To effectively use the context, fastText works better with sub-word level information, but we faced challenges in adapting the pre-processing (splitting words into sub-words), post-processing (combining sub-words into actual words), feature engineering (POS Tags, TF-IDF, TextRank etc.,), and calculating word-level attention over sub-words. With these added complexities during data processing and ML training, fastText is usually slower but it provides a rich context of the language.

In the view of the hierarchical-context, we need all the sentences of a single document at-a-time to calculate the attentions over words and sentences to create sentence-level and document-level vectors. To ease the calculation, we opted to use the dynamic batches for the data-loader i.e., a document with a different number of sentences will be sent into the encoder-decoder at once during the ML model training process. Ideally, we

choose a static number for the data-loader such as 64 samples for one iteration. Because of the dynamic nature of the data-loader, the number of iterations for the ML model training equals the number of documents shown, ultimately surging up the training time, making it 2-3 times slower than models without a hierarchical-context.

We also faced additional challenges with the newly and continuously added labels from the corpus in the fine-tuning process. For example, to engage broader HDEs, a crowd-sourced annotation approach is an excellent option to engage the wider community. However, identifying qualified HDEs conveniently to annotate labels is a complex task and brings auxiliary troubles.

6. Conclusion

This paper implements a hierarchical attention-driven KP identification model by retaining longer contextual dependencies and using minimal labeled data. It incrementally builds the context at the word and sentence levels across the document to understand the long-range context. The model demonstrates improved Accuracy for KP identification by adding document-level context through experimentation.

The domain adaptation in a semi-supervised approach also contributes to the creation of high-quality synthetic labels to solve the challenges with minimal labeled data. We have also found that the custom batch-loader yielding 2-4 true samples for every 100 synthetic samples helps the fine-tuning process with the limited labeled data and contributes to our understanding of optimizing the number of GS labeled-data required.

Finally, our methodology contributes to the general architectures of NLP in effectively creating ML models using limited labeled domain data by leveraging techniques of domain adaptation and document-level context, pre-trained LM, and pre-trained WE. Moreover, adding the character-level, text-based features to the model's encoder and confidence scores to the model's inference would further strengthen our results. These will be our following stage tasks, and we would like to experiment with them and publish the results shortly.

Acknowledgement(s)

The work was supported by the National Institute of General Medical Sciences of the National Institutes of Health under Award Number R01GM138589 and partially under P20GM121342.

We acknowledge Clemson University for the generous allotment of compute time on the Palmetto cluster for our experimentation. Also, we thank the anonymous reviewers for their detailed and insightful comments on earlier drafts of this paper.

Disclosure statement

There is no conflict of interest to disclose the details.

7. References

References

[1] Jing X, Min H, Gong Y, et al. A systematic review of ontology-based clinical decision support system rules: usage, management, and interoperability. medRxiv; 2022. https://doi.org/10.1101/2022.05.11.22274984.

[2] He Zhiyong, Wang Zanbo, Wei Wei, Feng Shanshan, Mao Xianling, Jiang Sheng. (2020). A Survey on Recent Advances in Sequence Labeling from Deep Learning Models. https://doi.org/10.48550/arXiv.2011.06727.

[3] Kazi Saidul Hasan, Vincent Ng. 2014. Automatic Keyphrase Extraction: A Survey of the State of the Art. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1262–1273, Baltimore, Maryland. Association for Computational Linguistics. http://dx.doi.org/10.3115/v1/P14-1119.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, L ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6000–6010. https://dl.acm.org/doi/10.5555/3295222.3295349.

[5] Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding https://doi.org/10.48550/arXiv.1810.04805.

[6] Zeyer Albert, Bahar Parnia, Irie Kazuki, Schluter Ralf, Ney Hermann. (2019). A Comparison of Transformer and LSTM Encoder Decoder Models for ASR. 8-15. https://doi.org/10.1109/ASRU46091.2019.9004025.

[7] Imran Ahamad Sheikh, Emmanuel Vincent, Irina Illina. Transformer versus LSTM Language Models Trained on Uncertain ASR Hypotheses in Limited Data Scenarios. LREC 2022 - 13th Language Resources and Evaluation Conference, Jun 2022, Marseille, France. hal-03362828v2

[8] Hochreiter S.„ Schmidhuber J"urgen. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[9] Lafferty J. D., McCallum A., Pereira F. C. N. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. Proceedings of the Eighteenth International Conference on Machine Learning (p./pp. 282–289), San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.. ISBN: 1-55860-778-1. https://dl.acm.org/doi/proceedings/10.5555/645530

[10] Sahrawat, D. et al. (2020). Keyphrase Extraction as Sequence Labeling Using Contextualized Embeddings. In: , et al. Advances in Information Retrieval. ECIR 2020. Lecture Notes in Computer Science(), vol 12036. Springer, Cham. https://doi.org/10.1007/978-3030-45442-5 41.

[11] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical Attention Networks for Document Classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1480–1489, San Diego, California. Association for Computational Linguistics. http://dx.doi.org/10.18653/v1/N16-1174

[12] Xu, Guohai & Wang, Chengyu & He, Xiaofeng. (2018). Improving Clinical Named Entity Recognition with Global Neural Attention: Second International Joint Conference, APWeb-WAIM 2018, Macau, China, July 23–25, 2018, Proceedings, Part II. 10.1007/9783-319-96893-3 20.

[13] Lobach D, Sanders G, Bright T, et al. Enabling Health Care Decision making Through Clinical Decision Support and Knowledge Management. Evidence Report No. 203. (Prepared by the Duke Evidence-based Practice Center under Contract No. 290-2007-10066-I.) AHRQ Publication No. 12-E001-EF.; 2012; Rockville, MD.

[14] Jing X, Himawan L, Law T. Availability and usage of clinical decision support systems (CDSS) in office-based primary care settings in the United States of America. BMJ Health & Care Informatics (under revision) 2019. https://doi.org/10.1136/bmjhci-2019-100015

[15] Salton, G; McGill, M. J. (1986). Introduction to modern information retrieval. McGrawHill. ISBN 978-0-07-054484-0. 10.5555/576628

[16] Hasan K. S., Ng V. (2010). Conundrums in unsupervised keyphrase extraction: Making sense of the state-of-the-art. In: Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, China, pp. 365–373. https://dl.acm.org/doi/proceedings/10.5555/1944566.

[17] Amati, G. (2009). BM25. In: LIU, L., OZSU, M.T. (eds) Encyclopedia of Database Sys-¨ tems. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-39940-9 921

[18] Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems, 30(1–7), 107–117. https://doi.org/10.1016/s0169-7552(98)00110-x

[19] Florian Boudin. 2018. Unsupervised Keyphrase Extraction with Multipartite Graphs. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pages 667–672, New Orleans, Louisiana. Association for Computational Linguistics. https://doi.org/10.48550/arXiv.1803.08721

[20] Corina Florescu and Cornelia Caragea. 2017. PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1105–1115, Vancouver, Canada. Association for Computational Linguistics. http://dx.doi.org/10.18653/v1/P17-1102

[21] Adrien Bougouin, Florian Boudin, and B´eatrice Daille. 2013. TopicRank: Graph-Based Topic Ranking for Keyphrase Extraction. In Proceedings of the Sixth International Joint Conference on Natural Language Processing, pages 543–551, Nagoya, Japan. Asian Federation of Natural Language Processing.

[22] Raschka, S. (2014). Naive Bayes and Text Classification I - Introduction and Theory. ArXiv, abs/1410.5329.

[23] Quinlan, J.R. Induction of decisiontrees. Mach Learn 1, 81–106 (1986). https://doi.org/10.1007/BF00116251

[24] Evgeniou, Theodoros & Pontil, Massimiliano. (2001). Support Vector Machines: Theory and Applications. 2049. 249-257. 10.1007/3-540-44673-7 12.

[25] Witten, Ian & Paynter, Gordon & Frank, Eibe & Gutwin, Carl & Nevill-Manning, Craig. (1999). KEA: Practical Automatic Keyphrase Extraction. ACM DL. 254-255. 10.1145/313238.313437.

[26] Chengzhi Zhang, Lei Zhao, Mengyuan Zhao, Yingyi Zhang. Enhancing Keyphrase Extraction from Academic Articles with their Reference Information. Scientometrics, 2022, 127(2): 703–731. https://doi.org/10.48550/arXiv.2111.14106

[27] Liu F., Pennell D., Liu F., Liu Y. (2009) Unsupervised approaches for automatic keyphrase extraction using meeting transcripts. In: Proceedings of Human Language Technologies: the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Boulder, Colorado, pp. 620–628. https://dl.acm.org/doi/proceedings/10.5555/1620754

[28] Xin Jiang, Yunhua Hu, and Hang Li. 2009. A ranking approach to keyphrase extraction. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval (SIGIR '09). Association for Computing Machinery, New York, NY, USA, 756–757. https://doi.org/10.1145/1571941.1572113

[29] Huang, Z.H., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. CoRR, abs/1508.01991 (2015)

[30] Jiang Y, Zhao T, Chai Y, et al. (2020) Bidirectional LSTM-CRF models for keyword extraction in Chinese sport news. In: MIPPR 2019: Pattern Recognition and Computer Vision. International Society for Optics and Photonics, 11430: 114300H. https://doi.org/10.48550/arXiv.1508.01991

[31] J. Li, A. Sun, J. Han and C. Li, "A Survey on Deep Learning for Named Entity Recognition," in IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 1, pp. 50-70, 1 Jan. 2022, doi: 10.1109/TKDE.2020.2981314. https://doi.org/10.1109/TKDE.2020.2981314

[32] Mikhailov, Vladislav & Shavrina, Tatiana. (2020). Domain-Transferable Method for Named Entity Recognition Task. 83-92. 10.5121/csit.2020.101407. https://doi.org/10.48550/arXiv.2011.12170

[33] Kulkarni, Vivek & Mehdad, Yashar & Chevalier, Troy. (2016). Domain Adaptation for Named Entity Recognition in Online Media with Word Embeddings. https://doi.org/10.48550/arXiv.1612.00148

[34] Honnibal, M., & Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

[35] Weischedel, Ralph, et al. OntoNotes Release 5.0 LDC2013T19. Web Download. Philadelphia: Linguistic Data Consortium, 2013. https://doi.org/10.35111/xmhb-2b84

[36] Christiane Fellbaum (1998, ed.) WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press. https://doi.org/10.7551/mitpress/7287.001.0001

[37] Jing X, Indani A, Hubig NC, et al. A systematic approach to configuring MetaMap for optimal performance. Methods Inf Med 2022 doi: 10.1055/a-1862-0421

[38] Neumann, M., King, D., Beltagy, I., & Ammar, W. (2019). ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. ArXiv, abs/1902.07669.

[39] Yoshua Bengio, R´ejean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. J. Mach. Learn. Res. 3, null (3/1/2003), 1137–1155. https://dl.acm.org/toc/jmlr/2003/3/null

[40] L. Rabiner and B. Juang, "An introduction to hidden Markov models," in IEEE ASSP Magazine, vol. 3, no. 1, pp. 4-16, Jan 1986, doi: 10.1109/MASSP.1986.1165342.

[41] Sachan, D.S., Xie, P., Sachan, M., & Xing, E.P. (2017). Effective Use of Bidirectional Language Modeling for Transfer Learning in Biomedical Named Entity Recognition. Machine Learning in Health Care. https://doi.org/10.48550/arXiv.1711.07908

[42] Lev Ratinov and Dan Roth. 2009. Design Challenges and Misconceptions in Named Entity Recognition. In Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009), pages 147–155, Boulder, Colorado. Association for Computational Linguistics. https://dl.acm.org/doi/proceedings/10.5555/1596374

[43] Saad, F., Aras, H., Hackl-Sommer, R. (2020). Improving Named Entity Recognition for Biomedical and Patent Data Using Bi-LSTM Deep Neural Network Models. In: M´etais, E., Meziane, F., Horacek, H., Cimiano, P. (eds) Natural Language Processing and Information Systems. NLDB 2020. Lecture Notes in Computer Science(), vol 12089. Springer, Cham. https://doi.org/10.1007/978-3-030-51310-8 3

[44] Mikolov, Tomas et al. "Efficient Estimation of Word Representations in Vector Space." International Conference on Learning Representations (2013). https://doi.org/10.48550/arXiv.1301.3781

[45] Enriching Word Vectors with Subword Information, Piotr Bojanowski, Edouard Grave, Armand Joulin and Tomas Mikolov, 2016. https://doi.org/10.48550/arXiv.1607.04606

[46] Bag of Tricks for Efficient Text Classification, Armand Joulin, Edouard Grave, Piotr Bojanowski, Tomas Mikolov, 2016. https://doi.org/10.48550/arXiv.1607.01759

[47] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, Doha, Qatar. Association for Computational Linguistics. http://dx.doi.org/10.3115/v1/D14-1162

[48] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," in IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673-2681, Nov. 1997, doi: 10.1109/78.650093.

[49] Luo L, Yang Z, Yang P, Zhang Y, Wang L, Lin H, Wang J. An attention-based BiLSTMCRF approach to document-level chemical named entity recognition. Bioinformatics. 2018 Apr 15;34(8):1381-1388. doi: 10.1093/bioinformatics/btx761. PMID: 29186323.

[50] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. arXiv preprint arXiv:1503.08895.

[51] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," in IEEE Transactions on Information Theory, vol. 13, no. 2, pp. 260-269, April 1967, doi: 10.1109/TIT.1967.1054010.

[52] Syed, Muzamil Hussain, and Sun-Tae Chung. 2021. "MenuNER: Domain-Adapted BERT Based NER Approach for a Domain with Limited Dataset and Its Application to Food Menu Domain" Applied Sciences 11, no. 13: 6007. https://doi.org/10.3390/app11136007

[53] Alexandra Schofield and David Mimno. 2016. Comparing Apples to Apple: The Effects of Stemmers on Topic Models. Transactions of the Association for Computational Linguistics, 4:287–300. http://dx.doi.org/10.1162/tacl_a_00099

[54] Benoit Favre. Contextual language understanding Thoughts on Machine Learning in Natural Language Processing. Computation and Language [cs.CL]. Aix-Marseille Universite, 2019. tel-02470185

[55] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. J. Mach. Learn. Res. 12, 2493– 2537 (2011). https://doi.org/10.48550/arXiv.1103.0398

[56] Benedict A. Rabut, Arnel C. Fajardo, and Ruji P. Medina. 2019. Multi-class Document Classification Using Improved Word Embeddings. In Proceedings of the 2nd International Conference on Computing and Big Data (ICCBD 2019). Association for Computing Machinery, New York, NY, USA, 42–46. https://doi.org/10.1145/3366650.3366661

[57] Nancy Chinchor and Beth Sundheim. 1993. MUC-5 Evaluation Metrics. In Fifth Message Understanding Conference (MUC-5): Proceedings of a Conference Held in Baltimore, Maryland, August 25-27, 1993.

[58] Hiroki Nakayama. "A Python framework for sequence labeling evaluation" 2018. https : //github.com/chakki – works/seqeval

8. Appendices

Appendix A. Dataset Details

Table A1. Showing explicit details of the CDSS dataset throughout the pre-processing steps

Type	Abstracts Number
Total after parsing PubMed XML	3326
HDE Labeled Set 1 (GS42)	42
ACM Abstracts [8] + HDE Labeled Set 2 (PMIDs not in XML) [4]	8+4 = 12
New Total with Duplicates (Some articles from GS42 are in Full Text XML)	3380
Abstracts (<3 sentences ~Little/No Abstract)	99
New Total with Duplicates	3281

(After removing abstracts with <3 sentences)	(1093 Train + 2188 Test)
HDE Labeled Set 2 (GS91) (ACM 8 + PubMed 83)	83 + 8 = 91
Total GS	91 + 42 = 133
Final Total (Synthetic Labeled dataset) (After removing GS 133 from Full Dataset)	3148 (1049 Train + 2099 Test)

Appendix B. Metrics for Fine-tuning on GS

Table B1. Fine-tune with GS Labels - 10 Experiments

GS	0	2	4	6	8	10	12
Precision	83.78 ± 11.12	86.43 ± 5.78	86.56 ± 9.86	84.32 ± 4.99	85.54 ± 6.94	84.35 ± 9.92	85.99 ± 10.75
Recall	80.88 ± 4.89	82.41 ± 7.85	82.26 ± 5.69	82.28 ± 2.21	82.91 ± 9.41	81.80 ± 15.32	82.99 ± 7.20
Accuracy	95.62 ± 0.93	96.21 ± 0.51	96.24 ± 1.04	95.65 ± 0.58	95.73 ± 1.12	95.88 ± 0.56	96.13 ± 0.80
F1-Score	82.44 ± 4.82	83.66 ± 3.79	84.48 ± 5.68	83.14 ± 2.05	83.35 ± 7.21	82.81 ± 4.38	84.22 ± 7.42

Table B2. Fine-tune with GS Labels - 50 Experiments

GS	0	2	4	6	8	10	12
Precision	83.78 ± 10.21	86.27 ± 8.28	86.29 ± 9.15	85.30 ± 8.01	85.95 ± 9.16	85.91 ± 8.70	86.22 ± 11.35
Recall	80.88 ± 4.49	83.16 ± 8.14	81.97 ± 11.91	82.53 ± 6.66	82.50 ± 7.83	82.45 ± 11.26	82.85 ± 8.20
Accuracy	95.62 ± 0.85	96.24 ± 0.74	96.16 ± 1.04	95.92 ± 0.72	96.06 ± 0.82	96.08 ± 0.87	96.27 ± 0.85
F1-Score	82.44 ± 4.43	84.65 ± 5.42	84.05 ± 7.36	83.65 ± 4.54	83.81 ± 6.05	83.91 ± 6.61	84.43 ± 6.59

Appendix C. Plots for Fine-tuning on GS

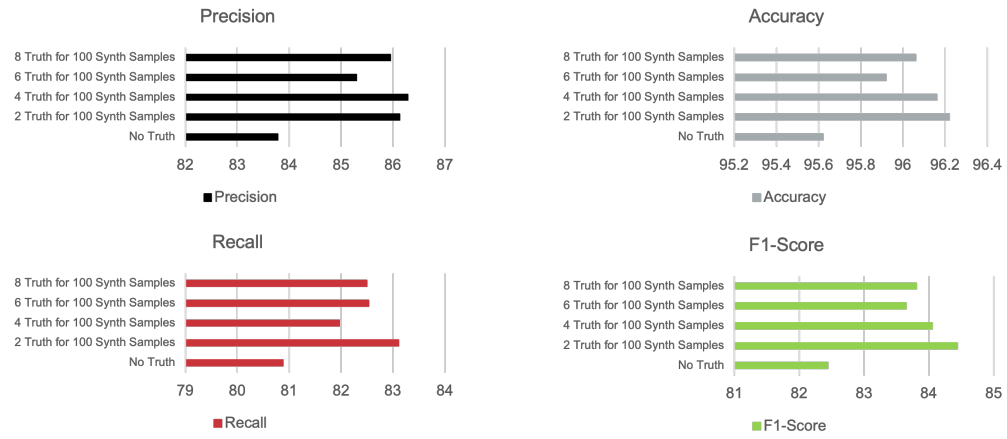

Figure C1. Plot evaluation metrics for fine-tuning with GS Labels - 50 Experiments

Figure C2. Plot evaluation metrics for fine-tuning with GS Labels - 10 Experiments