SurfCon: Synonym Discovery on Privacy-Aware Clinical Data

Zhen Wang*, Xiang Yue*, Soheil Moosavinasab†, Yungui Huang†, Simon Lin†, Huan Sun*
*The Ohio State University
{wang.9215,yue.149,sun.397}@osu.edu
†Abigail Wexner Research Institute at Nationwide Children's Hospital
{SeyedSoheil.Moosavinasab,Yungui.Huang,Simon.Lin}@nationwidechildrens.org

ABSTRACT

Unstructured clinical texts contain rich health-related information. To better utilize the knowledge buried in clinical texts, discovering synonyms for a medical query term has become an important task. Recent automatic synonym discovery methods leveraging raw text information have been developed. However, to preserve patient privacy and security, it is usually quite difficult to get access to large-scale raw clinical texts. In this paper, we study a new setting named *synonym discovery on privacy-aware clinical data* (i.e., medical terms extracted from the clinical texts and their aggregated co-occurrence counts, without raw clinical texts). To solve the problem, we propose a new framework SurfCon that leverages two important types of information in the privacy-aware clinical data, i.e., the *surface form information*, and the *global context information* for synonym discovery. In particular, the surface form module enables us to detect synonyms that look similar while the global context module plays a complementary role to discover synonyms that are semantically similar but in different surface forms, and both allow us to deal with the OOV query issue (i.e., when the query is not found in the given data). We conduct extensive experiments and case studies on publicly available privacy-aware clinical data, and show that SurfCon can outperform strong baseline methods by large margins under various settings.

KEYWORDS

Synonym Discovery, Privacy-Aware Clinical Data, Medical Term Recommendation

ACM Reference Format:
Zhen Wang*, Xiang Yue*, Soheil Moosavinasab†, Yungui Huang†, Simon Lin†, Huan Sun*. 2019. SurfCon: Synonym Discovery on Privacy-Aware Clinical Data . In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330894

1 INTRODUCTION

Clinical texts in Electronic Medical Records (EMRs) are enriched with valuable information including patient-centered narratives, patient-clinician interactions and disease treatment outcomes, which can be especially helpful for future decision making. To extract knowledge from unstructured clinical texts, synonym discovery [37] is an important task which can benefit many downstream applications. For example, when a physician issues a query term (e.g., "vitamin C") to find relevant clinical documents, automatically discovering its synonyms (e.g., "c vitamin", "vit c", "ascorbic acid") or even commonly misspelled variations (e.g. "viatmin c") can help to expand the query and thereby enhance the retrieval performance.

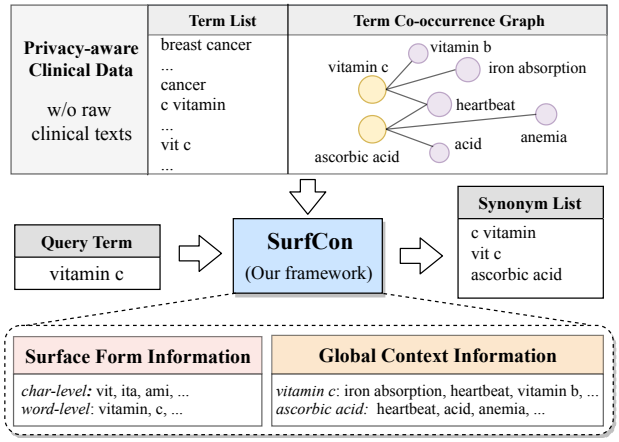


Figure 1: Task illustration: We aim to discover synonyms for a given query term from privacy-aware clinical data by effectively leveraging two important types of information: Surface form and global contexts.

For the sake of patient privacy and security, it is usually quite difficult, if not impossible, for medical institutes to grant public access to large-scale raw or even de-identified clinical texts [2]. Consequently, medical terms[1] and their aggregated co-occurrence counts extracted from raw clinical texts are becoming a popular (although not perfect) substitute for raw clinical texts for the research community to study EMR data [2, 8, 33]. For example, Finlayson et al. [8] released millions of medical terms extracted from the clinical texts in Stanford Hospitals and Clinics as well as their global co-occurrence counts, rather than releasing raw sentences/paragraphs/documents from the clinical text corpus. In this work, we refer to the given set of medical terms and their co-occurrence statistics in a clinical text corpus as *privacy-aware* clinical data, and investigate synonym discovery task on such data (Figure 1): *Given a set of terms extracted from clinical texts as well as their global co-occurrence graph*[2]*, recommend a list of synonyms for a query term.*

[1] A medical term is a single- or multi-word string (e.g., "Aspirin", "Acetylsalicylic Acid").
[2] where each node is a medical term and each edge between two nodes is weighted by the number of times that two terms co-occur in a given context window.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
KDD '19, August 4–8, 2019, Anchorage, AK, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330894

Developing effective approaches under this setting is particularly meaningful, as they will suggest that one can utilize less sensitive information (i.e., co-occurrence statistics rather than raw sentences in clinical texts) to perform the task well.

A straightforward approach to obtain synonyms is to map the query term to a knowledge base (KB) entity and retrieve its synonyms or aliases stored in the KBs. However, it is widely known that KBs are incomplete and outdated, and their coverage of synonyms can be very limited [38]. In addition, the informal writing of clinical texts often contain variants of surface forms, layman terms, frequently misspelling words, and locally practiced abbreviations, which should be mined to enrich synonyms in KBs. Recent works [30, 37, 42] have been focused on automatic synonym discovery from massive text corpora such as Wikipedia articles and PubMed paper abstracts. When predicting if two terms are synonyms or not, such approaches usually leverage the original sentences (a.k.a. *local* contexts) mentioning them, and hence do not apply or work well under our privacy-aware data setting where such sentences are unavailable.

Despite the lack of local contexts, we observe two important types of information carried in the privacy-aware data - surface form information and global context information (i.e., co-occurrence statistics). In this work, we aim to effectively leverage these two types of information for synonym discovery, as shown in Figure 1.

Some recent works [24, 25] model the similarity between terms in the character-level. For example, Mueller and Thyagarajan [24] learn the similarity between two sequences of characters, which can be applied for discovering synonyms that look alike such as "vit c" and "vitamin c". However, we observe two common phenomena that such approaches cannot address well and would induce false positive and false negative predictions respectively: (1) Some terms are similar in surface form but do not have the same meaning (e.g., "hemostatic" and "homeostasis", where the former means a process stopping bleeding while the latter refers to a constant internal environment in the human body); (2) Some terms have the same meaning but are different in surface form (e.g., "ascorbic acid" and "vitamin c" are the same medicinal product but look different).

On the other hand, given a term co-occurrence graph, various distributional embedding methods such as [18, 28, 34] have been proposed to learn a distributional representation (a.k.a. embedding) for each term based on its *global* contexts (i.e., terms connected to it in the co-occurrence graph). The main idea behind such methods is that two terms should have similar embedding vectors if they share a lot of global contexts. However, we observe that the privacy-aware clinical data tends to be very *noisy* due to the original data processing procedure[3], which presents new challenges for utilizing global contexts to model semantic similarity between terms. For example, Finlayson et al. [8] prune the edges between two terms co-occurring less than 100 times, which can lead to missing edges between two related terms in the co-occurrence graph. Ta et al. [33] remove all concepts with singleton frequency counts below 10. Hence, the noisy nature of the co-occurrence graph makes it less accurate to embed a term based on their original contexts. Moreover, when performing the synonym discovery task, users are very likely to issue a query term that does not appear in the given co-occurrence data. We refer to such query terms as Out-of-Vocabulary (OOV). Unlike In-Vocabulary[4] query terms, OOV query terms do not have their global contexts readily available in the given graph, which makes synonym discovery even more challenging.

In this paper, to address the above challenges and effectively utilize both the surface form and the global context information in the privacy-aware clinical data, we propose a novel framework named SurfCon which consists of a bi-level surface form encoding component and a context matching component, both based on neural models. The bi-level surface form encoding component exploits both character- and word-level information to encode a medical term into a vector. It enables us to compute a surface score of two terms based on their encoding vectors. As mentioned earlier, such surface score works well for detecting synonyms that look similar in surface form. However, it tends to miss synonymous terms that do not look alike. Therefore, we propose the context matching component to model the semantic similarity between terms, which plays a complementary role in synonymy discovery.

Our context matching component first utilizes the bi-level surface form encoding vector for a term to predict its potential global contexts. Using predicted contexts rather than the raw contexts in the given graph enables us to handle OOV query terms and also turns out to be effective for InV query terms. Then we generate a semantic vector for each term by aggregating the semantic features from predicted contexts using two mechanisms - static and dynamic representation mechanism. Specifically, given term a and term b, the dynamic mechanism aims to learn to weigh the importance of individual terms in a's contexts based on their semantic matching degree with b's contexts, while the static mechanism assigns equal weights to all terms in one's contexts. The former takes better advantage of individual terms within the contexts and empirically demonstrates superior performance.

Our contributions are summarized in three folds:

- We study the task of synonym discovery under a new setting, i.e., on privacy-aware clinical data, where only a set of medical terms and their co-occurrence statistics are given, and local contexts (e.g., sentences mentioning a term in a corpus) are not available. It is a practical setting given the wide concern about patient privacy for access to clinical texts and also presents unique challenges to address for effective synonym discovery.
- We propose a novel and effective framework named SurfCon that can discover synonyms for both In-Vocabulary (InV) and Out-of-Vocabulary (OOV) query terms. SurfCon considers two complementary types of information based on neural models - surface form information and global context information of a term, where the former works well for detecting synonyms that are similar in surface form while the latter can help better find synonyms that do not look alike but are semantically similar.
- We conduct extensive experiments on publicly available privacy-aware clinical data and demonstrate the effectiveness of our framework in comparison with various baselines and our own model variants.

[3]This tends to be a common issue in many scenarios as raw data has to go through various pre-processing steps for privacy concerns.

[4]Query terms that appear in the given co-occurrence graph are referred to as In-Vocabulary (InV).

2 TASK SETTING

In this section, we clarify several terminologies used in this paper as well as our problem definition:

Privacy-aware Clinical Data. Electronic medical records (EMRs) typically contain patient medical information such as discharge summary, treatment, and medical history. In EMRs, a significant amount of clinical information remains under-tapped in the unstructured clinical texts. However, due to privacy concerns, access to raw or even de-identified clinical texts in large quantities is quite limited. Also, traditional de-identification methods, e.g., removing the 18 HIPAA identifiers [32], require significant manual efforts for the annotation [7]. Moreover, there also exists the risk that de-identified data can be attacked and recovered by the re-identification in some cases [9]. Thus, to facilitate research on EMRs, an increasingly popular substitute strategy for releasing raw clinical texts is to extract medical terms and their aggregated co-occurrence counts from the corpus [2, 8, 33]. We refer to such data as privacy-aware clinical data in this paper. Converting raw sentences to co-occurrence data protects privacy as original patient records are very unlikely to be recovered. However, the local context information contained in the raw sentences is also lost, which makes various tasks including synonym discovery more challenging under privacy-aware datasets.

Medical Term Co-occurrence Graph. A medical term-term co-occurrence graph is defined as G=(V, E), where V is the set of vertices, each representing a medical term extracted from clinical texts. Each vertex has a surface form string (e.g., "vitamin c", "cancer") which is the spelling of the medical term. E is the set of edges, each weighted by how many times two terms co-occur in a certain context window (e.g., notes from patient records within 1 day).

Medical Term Synonym. Synonyms of a medical term refer to other medical terms that can be used as its alternative names [30]. For example, "vit c", "c vitamin" and "ascorbic acid" refer to the same medicinal product, while "Alzheimer's disease" and "senile dementia" represent the same disease. In our dataset, the extracted medical terms are mapped to the Unified Medical Language System (UMLS) [3] Concept Unique Identifier (CUI) by [8]. Different terms mapping to the same UMLS CUI are treated as synonyms for model training/development/testing.

Task Definition. We formally define our task of synonym discovery on privacy-aware clinical data as: *Given a medical term co-occurrence graph G, for a query term q (which can be either In-Vocabulary or Out-of-Vocabulary), recommend a list of medical terms from G that are likely to be synonyms of q.*

3 SURFCON FRAMEWORK

In this section, we introduce our proposed framework SurfCon for synonym discovery on privacy-aware clinical data.

3.1 Overview

We observe two important types of information carried in the privacy-aware clinical data: surface form information of a medical term and the global contexts from the given co-occurrence graph. On the one hand, existing approaches [25] using character-level features to detect synonyms could work well when synonyms share a high string similarity, but tend to produce false positive predictions (when two terms look similar but are not synonyms, e.g., "hemostatic" and "homeostasis") and false negative predictions (when two terms are synonyms but look very different, e.g., "ascorbic acid" and "vitamin c"). On the other hand, the global contexts of a term under the privacy-aware setting tend to be noisy partly due to the original data pre-processing procedure, which also presents challenges for using them to model the semantic similarity between terms. Thus, a framework that is able to effectively leverage these two types of information needs to be carefully designed.

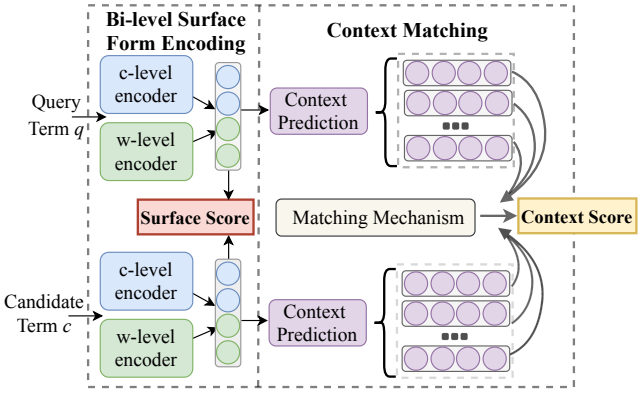


Figure 2: Framework overview. For each query term, a list of candidate terms will be ranked based on both the surface and context scores.

Towards that end, we propose SurfCon (Figure 2) and summarize its high-level ideas as below:

(1) Given a query term (whether being InV or OOV), the bi-level surface form encoding component and the context matching component score a candidate term[5] respectively based on the surface form information and global context information. The former enables us to find synonyms that look similar to the query term by considering both character- and word-level information, and the latter complements it by capturing the semantic similarity between terms to better address the false positive and false negative problem mentioned earlier.

(2) Considering the original global contexts being noisy as well as the existence of OOV query terms, instead of directly leveraging the raw global contexts, the context matching component will first utilize the surface form encoding vector of a term to *predict* its potential global contexts[6]. We then investigate a novel dynamic context matching mechanism (see Section 3.2.2 for details) to evaluate if two terms are synonyms based on their predicted contexts.

(3) The two components are combined by a weighted score function, in which parameters are jointly optimized with a widely used ranking algorithm ListNet [5]. At testing time, given a query term, candidate terms are ranked based on the optimized score function.

3.2 Methodology

Now we describe the two components of SurfCon: Bi-level Surface Form Encoding and Context Matching in details.

3.2.1 ***Bi-level Surface Form Encoding.*** The bi-level surface form encoding of our framework aims to model the similarity between two terms at the surface form level, as we observe that two terms

[5]Every term in the given co-occurrence graph can be a candidate term.

[6]For terms in the co-occurrence graph, predicting contexts can be treated as denoising its original global contexts (or edges)

tend to be synonymous if they are very similar in surface forms. Such observation is intuitive but works surprisingly well in synonym discovery task. Driven by this observation, we design the bi-level surface form encoding component in a way that both of character- and word-level information of terms are captured. Then, a score function is defined to measure the surface form similarity for a pair of terms based on their bi-level encoding vectors. The bi-level encoders are able to encode surface form information of both InV terms and OOV terms.

Specifically, as shown in Figure 2, given a query term q and a candidate term c, we denote their character-level sequences as $x_q = \{x_{q,1}, ..., x_{q,m_q}\}, x_c = \{x_{c,1}, ..., x_{c,m_c}\}$, and their word-level sequences as $w_q = \{w_{q,1}, ..., w_{q,n_q}\}, w_c = \{w_{c,1}, ..., w_{c,n_c}\}$, where m_q, n_q, m_c, n_c are the length of the character-level sequence and word-level sequence of the query term and the candidate term respectively. Then we build two encoders ENC^{ch} and ENC^{wd} to capture the surface form information at the character- and word-level respectively:

$$\begin{aligned} s_q^{ch} &= \text{ENC}^{ch}(x_{q,1}, \ldots, x_{q,m_q}),\ s_q^{wd} = \text{ENC}^{wd}(w_{q,1}, \ldots, w_{q,n_q}) \\ s_c^{ch} &= \text{ENC}^{ch}(x_{c,1}, \ldots, x_{c,m_c}),\ s_c^{wd} = \text{ENC}^{wd}(w_{c,1}, \ldots, w_{c,n_c}) \end{aligned} \tag{1}$$

where $s_q^{ch}, s_c^{ch} \in \mathbb{R}^{d_c}$ are the character-level embeddings for the query and candidate terms, and $s_q^{wd}, s_c^{wd} \in \mathbb{R}^{d_w}$ are the word-level embeddings for the query and candidate terms respectively.

Note that there has been a surge of effective encoders that model sequential information from character-level or word-level, ranging from simple look-up table (e.g., character n-gram [13] and Skip-Gram [23]) to complicated neural network architectures (e.g., CNN [14], LSTM [1] and Transformer [35], etc.). For simplicity, here, we adopt simple look-up tables for both character-level embeddings and word-level embeddings. Instead of randomly initializing them, we borrow pre-trained character n-gram embeddings from Hashimoto et al. [13] and word embeddings from Pennington et al. [28]. Our experiments also demonstrate that these simple encoders can well encode surface form information of medical terms for synonym discovery task. We leave evaluating more complicated encoders as our future work.

After we obtain the embeddings at both levels, we concatenate them and apply a nonlinear function to get the surface vector s for the query and candidate term. Let us denote such encoding process as a function $h(\cdot)$ with the input as term q or c and the output as the surface vector s_q or s_c:

$$\begin{aligned} s_q &= h(q) = \tanh([s_q^{ch}, s_q^{wd}]W_s + b_s), \\ s_c &= h(c) = \tanh([s_c^{ch}, s_c^{wd}]W_s + b_s) \end{aligned} \tag{2}$$

where the surface vectors $s_q, s_c \in \mathbb{R}^{d_s}$, and $W_s \in \mathbb{R}^{(d_c+d_w)\times d_s}, b_s \in \mathbb{R}^{d_s}$ are weight matrix and bias for a fully-connected layer.

Next, we define the surface score for a query term q and a candidate term c to measure the surface form similarity based on their encoding vectors s_q and s_c:

$$\text{Surface Score}\,(q, c) = f_s(s_q, s_c) \tag{3}$$

3.2.2 ***Context Matching.*** In order to discover synonyms that are not similar in surface form, and also observing that two terms tend to be synonyms if their global contexts in the co-occurrence graph are semantically very relevant, we design the context matching component to capture the semantic similarity of two terms by carefully leveraging their global contexts. We first illustrate the intuition behind this component using a toy example:

Example 1. **[Toy Example for Illustration.]** *Assume we have a query term "vitamin c" and a candidate term "ascorbic acid". The former is connected with two terms "iron absorption" and "vitamin b" in the co-occurrence graph as global contexts, while the latter has "fatty acids" and "anemia" as global contexts.*

Our context matching component essentially aims to use a term's contexts to represent its semantic meaning and a novel *dynamic context matching mechanism* is developed to determine the importance of each individual term in one's contexts. For example, *"iron absorption"* is closely related to *"anemia"* since the disease "anemia" is most likely to be caused by the iron deficiency. Based on the observation, we aim to increase the relative importance of *"iron absorption"* and *"anemia"* in their respective context sets when representing the semantic meaning of *"vitamin c"* and *"ascorbic acid"*. Therefore, we develop a novel dynamic context matching mechanism to be introduced shortly.

In order to recover global contexts for OOV terms and also noticing the noisy nature of the co-occurrence graph mentioned earlier, we propose an *inductive context prediction module* to predict the global contexts for a term based on its surface form information instead of relying on the raw global contexts in the given co-occurrence graph.

Inductive Context Prediction Module. Let us first denote a general medical term as t. For a term-term co-occurrence graph, we treat all InV terms as possible context terms and denote them as $\{u_j\}_{j=1}^{|V|}$ where $|V|$ is the total number of terms in the graph. The inductive context prediction module aims to predict how likely term u_j appears in the context of t (denoted as the conditional probability $p(u_j|t)$). To learn a good context predictor, we utilize all existing terms in the graph as term t, i.e., $t \in \{u_i\}_{i=1}^{|V|}$ and the conditional probability becomes $p(u_j|u_i)$.

Formally, the probability of observing term u_j in the context of term u_i is denoted as:

$$p(u_j|u_i) = \frac{\exp(v_{u_j}^T \cdot s_{u_i})}{\sum_{k=1}^{|V|} \exp(v_{u_k}^T \cdot s_{u_i})} \tag{4}$$

where $s_{u_i} = h(u_i)$ and $h(\cdot)$ is the same encoder function defined in section 3.2.1. $v_{u_j} \in \mathbb{R}^{d_o}$ is the context embedding vector corresponding to term u_j and we let $d_o = d_s$. The predicted distribution $p(u_j|u_i)$ is optimized to be close to the empirical distribution $\hat{p}(u_j|u_i)$ defined as:

$$\hat{p}(u_j|u_i) = \frac{w_{ij}}{\sum_{(i,k)\in E} w_{ik}} \tag{5}$$

where E is the set of edges in the co-occurrence graph and w_{ij} is the weight between term u_i and term u_j. We adopt the cross entropy loss function for optimizing:

$$L_n = -\sum_{u_i, u_j \in V} \hat{p}(u_j|u_i) \log(p(u_j|u_i)) \tag{6}$$

When the number of terms in the graph $|V|$ is very large, it is computationally costly to calculate the conditional probability $p(u_j|u_i)$, and one can utilize the negative sampling algorithm [22]

to train our inductive context predictor efficiently. The loss function Eqn. 6 can be modified as:

$$\log \sigma(v_{u_j}^T \cdot s_{u_i}) + \sum_{n=1}^{N_0} E_{u_n \sim P_n(u)}[\log \sigma(-v_{u_n}^T \cdot s_{u_i})] \quad (7)$$

where $\sigma(x) = 1/(1+\exp(-x))$ and u_n is the negative sample drawn from the noise distribution $P_n(u) \propto d_u^{3/4}$. N_0 is the number of negative samples and d_u is the degree of term u in the co-occurrence graph.

Now, given a term t (either InV or OOV), we can select the top-K terms as its predicted contexts based on the predicted probability distribution $p(\cdot|t)$. Next, we describe the dynamic context matching mechanism to model the semantic similarity of two terms based on their predicted contexts.

Dynamic Context Matching Mechanism. Inspired by previous works on neighborhood aggregation based graph embedding methods [12, 36], which generate an embedding vector for an InV node by aggregating features from its neighborhood (contexts), we introduce two semantic vectors respectively for the query term and the candidate term, $v_q, v_c \in \mathbb{R}^{d_e}$, and learn them by aggregating the feature vectors of their corresponding top-K predicted contexts from previous module.

Let us define $v_q^i \in \mathbb{R}^{d_e}$ as the feature vector of the i-th term in query term q's context while $v_c^j \in \mathbb{R}^{d_e}$ as the feature vector of the j-th term in candidate term c's context, and their context sets as $\Phi(q) = \{v_q^i\}_{i=1}^K$, $\Phi(c) = \{v_c^j\}_{j=1}^K$. Essentially, as we aim to capture the semantic meaning of terms, the feature vectors v_q^i's and v_c^j's are expected to contain semantic information. Also noticing that all predicted context terms are InV terms (i.e., in the co-occurrence graph), which allows us to adopt widely used graph embeddings, such as LINE(2nd) [34] as their feature vectors.

One naive way to obtain the context semantic vectors, v_q and v_s is to average vectors in their respective context set. Since such v_q (or v_c) does not depend on the other one, we refer to such vectors as "static" representations for terms.

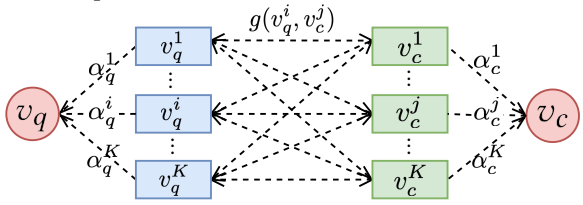


Figure 3: Dynamic Context Matching Mechanism.

In contrast to the static approach, we propose the *dynamic context matching mechanism* (as shown in Figure 3), which weighs each term in the context of q (or c) based on its matching degree with terms in the context of c (or q) and hence the context semantic vector representation v_q (or v_c) is *dynamically* changing depending on which terms it is comparing with. More specifically, let us define $g(x, y) = \tanh(xW_m y^T)$ as a nonlinear function parameterized with weight matrix $W_m \in \mathbb{R}^{d_e \times d_e}$ to measure the similarity between two row vectors x and y. For each context vector v_q^i of the query term, we calculate its weight based on how it matches with c's contexts overall:

$$\text{match}[v_q^i, \Phi(c)] = \text{Pooling}[g(v_q^i, v_c^1), ..., g(v_q^i, v_c^K)] \quad (8)$$

For the pooling operation, we empirically choose the mean pooling strategy as it performs better than alternatives such as max pooling in our experiments. Then we normalize the weight of v_q^i as:

$$\alpha_q^i = \frac{e^{\text{match}[v_q^i, \Phi(c)]}}{\sum_{k=1}^K e^{\text{match}[v_q^k, \Phi(c)]}} \quad (9)$$

Finally, the context semantic vector for the query term v_q is calculated through a weighted combination of q's contexts:

$$v_q = \sum_{i=1}^K \alpha_q^i \cdot v_q^i \quad (10)$$

Following the same procedure, we can obtain the context semantic vector v_c for the candidate term w.r.t. the query term. Then we define the context score for a query term q and a candidate term c to measure their semantic similarity based on v_q and v_c:

$$\text{Context Score}(q, c) = f_c(v_q, v_c) \quad (11)$$

3.3 Model Optimization and Inference

Objective Function. Given a query term q and a candidate term c, to capture their similarity based on surface forms and global contexts, we define the final score function as:

$$f(q, c) = (1-\gamma) \cdot f_s(s_q, s_c) + \gamma \cdot f_c(v_q, v_c) \quad (12)$$

$f_s(\cdot)$ and $f_c(\cdot)$ are similarity functions between two vectors, e.g., cosine similarity or bilinear similarity. Now we obtain the recommendation probability of each candidate $t_i \in \{t_1, ..., t_N\}$ given a query q:

$$p(t_i|q) = \frac{e^{f(q,t_i)}}{\sum_{k=1}^N e^{f(q,t_k)}} \quad (13)$$

where N is the size of the candidate set. Finally, we adopt the ListNet [5] ranking framework which minimizes the cross entropy loss for query term q:

$$L_r = -\sum_{i=1}^N p^*(t_i|q) \log p(t_i|q) \quad (14)$$

where $p^*(t_i|q)$ is the normalized ground-truth distribution of a list of ranking scores as $\{r_i\}_{i=1}^N$ where r_i equals to 1 if q and t_i are synonyms and 0 otherwise.

Training. For efficiency concerns, we adopt a two-phase training strategy: We first train the inductive context prediction module by loss function L_n (Eqn. 6) in the term-term co-occurrence graph, and sample top-K contexts based on the predicted probability distribution and use them in the context matching component. Then, we train the ranking framework by minimizing the ranking loss L_r (Eqn. 14).

Inference. At the inference stage, we treat all InV terms as candidates for a given query. Since the dynamic representation mechanism involves pairwise term matching between the contexts of the query term and those of each candidate term and can have a high computational cost when the candidate set size is large, we adopt a two-step strategy: (1) For a given query term, select its top-N high potential candidates based on the surface form encoding vector and the context semantic vector obtained by the static representation mechanism; (2) Re-rank the selected candidates by applying our SurfCon framework with the dynamic representation mechanism.

4 EXPERIMENTS

Now we evaluate our proposed framework SurfCon to show the effectiveness of leveraging both surface form information and global context information for synonym discovery.

4.1 Datasets

Medical Term Co-occurence Graph. We adopt publicly available sets of medical terms with their co-occurrence statistics which are extracted by Finlayson et al. [8] from 20 million clinical notes collected from Stanford Hospitals and Clinics[20] since 1995. Medical terms are extracted using an existing phrase mining tool [16] by matching with 22 clinically relevant ontologies such as SNOMED-CT and MedDRA. And co-occurrence frequencies are counted based on how many times two terms co-occur in the same temporal *bin* (i.e., a certain timeframe in patient's records), e.g., 1, 7, 30, 90, 180, 365, and ∞-day *bins*.

Without loss of generality, we choose 1-day per-bin and ∞-day per-bin[7] graphs to evaluate different methods. We first convert the global counts between nodes to the PPMI values [17] and adopt subsampling [23] to filter very common terms, such as "medical history", "medication dose", etc. We choose these two datasets because they have very different connection density as shown in Table 1, and denote them as **1-day** and **All-day** datasets.

Synonym Label. In the released datasets, Finlayson et al. [8] provided a term-to-UMLS CUI mapping based on the same 22 ontologies as used when extracting terms. They reduced the ambiguity of a term by suppressing its least likely meaning so as to provide a high-quality mapping. We utilized such mapping to obtain the synonym labels: Terms mapped to the same UMLS CUI are treated as synonyms, e.g., terms like "c vitamin", "vit c", "ascorbic acid" are synonyms as they are all mapped to the concept "Ascorbic Acid" with ID C0003968.

Query Terms. Given a medical term-term co-occurrence graph, terms in the graph that can be mapped to UMLS CUIs are treated as potential query terms, and we split all such terms into training, development and testing sets. Here, since all terms appear in the given co-occurrence graph, this testing set is referred to as the **InV testing set**. We also create an **OOV testing set**: Under a UMLS CUI, terms not in the co-occurrence graph are treated as OOV query terms and are paired with their synonyms which are in the graph to form positive pairs. We sample 2,000 of such OOV query terms for experiments. In addition, since synonyms with different surface forms tend to be more challenging to discover (e.g., "vitamin c" vs. "ascorbic acid"), we also sample a subset named **Dissim** under both InV and OOV testing set, where query terms paired with their dissimilar synonyms[8] are selected. Statistics of our training/dev/testing sets are given in Table 1.

4.2 Experimental Setup

4.2.1 Baseline methods. We compare SurfCon with the following 10 methods. The baselines can be categorized by three types: (i) Surface form based methods, which focus on capturing the surface form information of terms. (ii) Global context based methods, which try to learn embeddings of terms for synonym discovery; (iii) Hybrid methods, which combine surface form and global context information. The others are our model variants.

[7]Per-bin means each unique co-occurring term-term pair is counted at most once for each relevant bin of a patient. We refer readers to Finlayson et al. [8] for more information.

[8]Dissimilarity is measured by Levenshtein edit distance [10] with a threshold (0.8).

Table 1: Statistics of our datasets.

		1-day dataset	All-day dataset
# Nodes		52,804	43,406
# Edges		16,197,319	50,134,332
Average # Degrees		613.5	2310.0
# Train Terms		9,451	7,021
# Dev Terms		960	726
# InV Test Terms	All	960	726
	Dissim	175	152
# OOV Test Terms	All	2,000	2,000
	Dissim	809	841

Surface form based methods. (1) *CharNgram* [13]: We borrow pre-trained character n-gram embeddings from Hashimoto et al. [13] and take the average of unique n-gram embeddings for each term as its feature, and then train a bilinear scoring function following previous works [30, 42]. (2) *CHARAGRAM* [40]: Similar as above, but we further fine-tune CharNgram embeddings using synonym supervision. (3) *SRN* [25]: A Siamese network structure is adopted with a bi-directional LSTM to encode character sequence of each term and cosine similarity is used as the scoring function.

Global context based methods. (4) *Word2vec* [23]: A popular distributional embedding method. We obtain word2vec embeddings by doing SVD decomposition over the Shifted PPMI co-occurrence matrix [18]. We treat the embeddings as features and use a bilinear score function for synonym discovery. (5) *LINE(2nd)* [34]: A widely-adopted graph embedding approach. Similarly, embeddings are treated as features and a bilinear score function is trained to detect synonyms. (6) *DPE-NoP* [30]: DPE is proposed for synonym discovery on text corpus, and consists of a distributional module and a pattern module, where the former utilizes global context information and the latter learns patterns from raw sentences. Since raw texts are unavailable in our setting, we only deploy the distributional module (a.k.a. DPE-NoP in Qu et al. [30]).

Hybrid methods. (7) *Concept Space Model* [37]: A medical synonym extraction method that combines word embeddings and heuristic rule-based string features. (8) *Planetoid* [41]: An inductive graph embedding method that can generate embeddings for both observed and unseen nodes. We use the bi-level surface form encoding vectors as the input and take the intermediate hidden layer as embeddings. Similarly, a bilinear score function is used for synonym discovery.

Model variants. (9) *SurfCon (Surf-Only)*: A variant of our framework which only uses the surface score for ranking. (10) *SurfCon (Static)*: Our framework with static representation mechanism. By comparing these variants, we verify the performance gain brought by modeling global contexts using different matching mechanisms.

For baseline methods (1-3 and 8) and our models, we test them under both InV and OOV settings. For the others (4-7), because they rely on embeddings that are only available for InV terms, we only test them under InV setting.

4.2.2 Candidate Selection and Performance Evaluation. For evaluating baseline methods and our model, we experiment with two strategies: (1) Random candidate selection. For each query term, we randomly sample 100 non-synonyms as negative samples and mix

Table 2: Model evaluation in MAP with random candidate selection.

Method Category	Methods	**1-day** Dataset					**All-day** Dataset				
		Dev	InV Test		OOV Test		Dev	InV Test		OOV Test	
			All	Dissim	All	Dissim		All	Dissim	All	Dissim
Surface form based methods	CharNgram [13]	0.8755	0.8473	0.4657	0.7427	0.4131	0.8652	0.8553	0.4615	0.7675	0.4424
	CHARAGRAM [40]	0.8705	0.8507	0.5504	0.7609	0.5142	0.8915	0.8805	0.5153	0.8119	0.5282
	SRN [25]	0.8886	0.8565	0.5102	0.7241	0.4341	0.8460	0.8170	0.4523	0.7110	0.4176
Global context based methods	Word2vec [23]	0.3838	0.3748	0.3188	-	-	0.4801	0.476	0.4180	-	-
	LINE(2nd) [34]	0.4279	0.4301	0.3494	-	-	0.5068	0.5043	0.4369	-	-
	DPE-NoP [30]	0.6222	0.6107	0.4855	-	-	0.5928	0.5949	0.4938	-	-
Hybrid methods (surface+context)	Concept Space [37]	0.8094	0.8109	0.4690	-	-	0.8064	0.7924	0.5574	-	-
	Planetoid [41]	0.8813	0.8514	0.5612	0.731	0.4714	0.8818	0.8765	0.6963	0.7403	0.4986
Our model and variants	SurfCon (Surf-Only)	0.9160	0.9053	0.6145	0.8228	0.5829	0.9034	0.8958	0.6006	0.8183	0.5622
	SurfCon (Static)	0.9242	0.9151	0.6542	0.8285	0.5933	0.9170	0.9019	0.6656	0.8203	0.5664
	SurfCon	**0.9348**	**0.9176**	**0.6821**	**0.8301**	**0.6009**	**0.9219**	**0.9199**	**0.7171**	**0.8232**	**0.5673**

them with synonyms for testing. This strategy is widely adopted by previous work on synonym discovery for testing efficiency [37, 42]. (2) Inference-stage candidate selection. As mentioned in section 3.3, at the inference stage, we first obtain high potential candidates in a lightweight way. Specifically, after the context predictor is pre-trained, for all terms in the given graph as well as the query term, we generate their surface form vector s and context semantic vector v obtained by the static representation. Then we find top 50 nearest neighbors of the query term respectively based on s and v using cosine similarity. Finally, we apply our methods and baselines to re-rank the 100 high potential candidates. We refer to these two strategies as *random candidate selection* and *inference-stage candidate selection*.

For evaluation, we adopt a popular ranking metric Mean Average Precision defined as $\text{MAP} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{m_i}\sum_{j=1}^{m_i}\text{Precision}(R_{ij})$, where R_{ij} is the set of ranked terms from 1 to j, m_i is the length of i-th list, and $|Q|$ is the number of queries.

4.2.3 Implementation details. Our framework is implemented in Pytorch [27] with Adam optimizer [15]. The dimensions of character embeddings (d_c), word embeddings (d_w), surface vectors (d_s), and sementic vectors (d_e) are set to be 100, 100, 128, 128. Early stopping is used when the performance in the dev sets does not increase continuously for 10 epochs. We directly optimize Eqn. 6 since the number of terms in our corpus is not very large, and set $f_s(\cdot)$ and $f_c(\cdot)$ to be cosine similarity and bilinear similarity function respectively, based on the model performance on the dev sets. When needed, string similarities are calculated by using the Distance package[9]. Pre-trained CharNgram [13] embeddings are borrowed from the authors[10]. For CHARAGRAM [40], we initialize the n-gram embeddings by using pre-trained CharNgram and fine-tune them on our dataset by the synonym supervision. We learn LINE(2nd) embeddings [34] by using OpenNE[11]. Heuristic rule-based matching features of Concept Space model are implemented according to [37]. Code, datasets, and more implementation details are available online[12].

[9]https://github.com/doukremt/distance
[10]https://github.com/hassyGo/charNgram2vec
[11]https://github.com/thunlp/OpenNE
[12]https://github.com/yzabc007/SurfCon

4.3 Results and Analysis

4.3.1 Evaluation with Random Candidate Selection. We compare all methods under random candidate selection strategy with the results shown in Table 2.

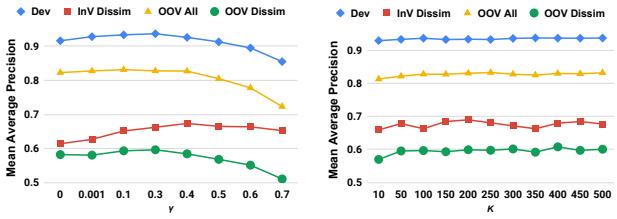
(1) Comparing SurfCon with surface form based methods. Our model beats all surface form based methods, including strong baselines such as SRN that use complicated sequence models to capture character-level information. This is because: 1) Bi-level encoder of SurfCon could capture surface form information from both character- and word-level, while baselines only consider either of them; 2) SurfCon captures global context information, which could complement surface form information for synonym discovery. In addition, in comparison with CharNgram and CHARAGRAM, our model variant SurfCon (Surf-Only), which also only uses surface form information, obtains consistently better performance, especially in the OOV Test set. The results demonstrate that adding word-level surface form information is useful to discover synonyms.

(2) Comparing SurfCon with global context based methods. SurfCon substantially outperforms all other global context based methods (Word2vec, LINE(2nd) and DPE-NoP). This is largely due to the usage of surface form information. In fact, as one can see, global context based methods are generally inferior to surface form based methods, partly due to the fact that a large part of synonyms are similar in surface form, while only a small portion of them are in very different surface form. Thus, detecting synonyms without leveraging surface information can hardly lead to good results. Besides, our context matching component conducts context prediction and matching strategies, which takes better advantage of global context information and thus lead to better performance on the synonym discovery task.

(3) Comparing SurfCon with hybrid methods. We also compare our model with baselines that combine both surface form and global context information. First, SurfCon is superior to the concept space model because the latter simply concatenates distributional embeddings with rule-based string features, e.g., the number of shared words as features and apply a logistic regression classifier for classification. Further, SurfCon also performs better than Planetoid, partly because our framework more explicitly leverages both surface form and global context information to formulate

Table 3: Model evaluation at inference stage.

Methods	1-day		All-day	
	InV Test	OOV Test	InV Test	OOV Test
CHARAGRAM [13]	0.3921	0.4044	0.3941	0.3913
DPE-NoP [30]	0.2396	-	0.2408	-
Planetoid [41]	0.4563	0.4268	0.3765	0.3812
SurfCon	0.5525	0.5068	0.4686	0.4661



(a) Coefficient of context score (b) Number of contexts

Figure 4: Performance w.r.t. (a) the coefficient of context score γ and (b) the number of context terms K.

synonym scores, while Planetoid relies on one embedding vector for each term which only uses surface form information as input.

(4) Comparing SurfCon with its variants. To better understand why SurfCon works well, we compare it with several variants. Under both datasets, SurfCon (Surf-Only) already outperforms all baselines demonstrating the effectiveness of our bi-level surface form encoding component. With the context matching component in SurfCon (Static), the performance is further improved, especially under *InV Test Dissim* setting where synonyms tend to have different surface forms and we observe around 4% performance gain. Further, by using dynamic representation in context matching mechanism, SurfCon obtains better results, which demonstrates that the dynamic representation is more effective to utilize context information compared with the static strategy.

4.3.2 Evaluation at Inference Stage. To further evaluate the power of our model in real practice, we test its performance at the inference stage as mentioned in section 3.3. Due to space constraint, we only show the comparison in Table 3 between SurfCon and several strong baselines revealed by Table 2. In general, the performance of all methods decreases at the inference stage compared with the random candidate selection setting, because the constructed list of candidates becomes harder to rank since surface form and context information are already used for the construction. For example, a lot of non-synonyms with similar surface form are often included in the candidate list. Even though the task becomes harder, we still observe our model outperforms the strong baselines by a large margin (e.g., around 8% at least) under all settings.

4.3.3 Parameter Sensitivity. Here we investigate the effect of two important hyper-parameters: The coefficient γ which balances the surface score and the context score, and the number of predicted contexts K used for context matching. As shown in Figure 4(a), the performance of SurfCon first is improved as γ increases, which is expected because as more semantic information is incorporated, SurfCon could detect more synonyms that are semantically similar. When we continue to increase γ, the performance begins to decrease and the reason is that surface form is also an important source of information that needs to be considered. SurfCon achieves the best performance roughly at $\gamma = 0.3$ indicating surface form information is relatively more helpful for the task than global context information. This also aligns well with our observation that synonyms more often than not have similar surface forms. Next, we show the impact of K in Figure 4(b). In general, when K is small (e.g., $K = 10$), the performance is not as good since little global context information is considered. Once K increases to be large enough (e.g., ≥ 50), the performance is not sensitive to the variation under most settings showing that we can choose smaller K for computation efficiency but still with good performance.

Table 4: Case studies on the 1-day dataset. Bold terms are synonyms in our labeled set while underlined terms are not but quite similar to the query term in semantics.

Query Term	"unable to vocalize" (InV)	"marijuana" (OOV)
SurfCon Top Ranked Candidates	"does not vocalize" "aphonia" "loss of voice" "vocalization" **"unable to phonate"**	**"marijuana abuse"** **"cannabis"** "cannabis use" "marijuana smoking" "narcotic"
Labeled Synonym Set	"unable to phonate"	"cannabis" "marijuana abuse" "marihuana abuse"

4.4 Case Studies

We further conduct case studies to show the effectiveness of SurfCon. Two query terms "unable to vocalize" and "marijuana" are chosen respectively from the InV and OOV test set where the former is defined as the inability to produce voiced sound and the latter is a psychoactive drug used for medical or recreational purposes. As shown in Table 4, for the InV query "unable to vocalize", our model can successfully detect its synonyms such as "unable to phonate", which already exists in the labeled synonym set collected based on term-to-UMLS CUI mapping as we discussed in Section 2. More impressively, our framework also discovers some highly semantically similar terms such as "does not vocalize" and "aphonia", even if some of them are quite different in surface form from the query term. For the OOV query "marijuana", SurfCon ranks its synonym "marijuana abuse" and "cannabis" at a higher place. Note that the other top-ranked terms are also very relevant to "marijuana".

5 RELATED WORK

Character Sequence Encoding. To capture the character-level information of terms, neural network models such as Recurrent Neural Networks and Convolutional Neural Networks can be applied on character sequences [1, 14]. Further, CHARAGRAM [40], FastText [4], and CharNGram [13] are proposed to represent terms and their morphological variants by capturing the shared subwords and n-grams information. However, modeling character-level sequence information only is less capable of discovering semantically similar synonyms, and our framework considers global context information to discover those synonyms.

Word and Graph/Network Embedding. Word embedding methods such as word2vec [23] and Glove [28] have been proposed and successfully applied to mining relations of medical phrases [26, 37]. More recently, there has been a surge of graph embedding methods that seek to encode structural graph information into low-dimensional dense vectors, such as Deepwalk [29], LINE [34]. Most of the embedding methods can only learn embedding vectors for

words in the corpus or nodes in the graph, and thus fail to address the OOV issue. On the other hand, some more recent inductive graph embedding works, such as Planetoid [41], GraphSAGE [12], and SEANO [19], could generate embeddings for nodes that are unobserved in the training phase by utilizing their node features (e.g., text attributes). *However, most of them assume the neighborhood of those unseen nodes is known, which is not the case for our OOV issue as the real contexts of an OOV term are unknown.* Since Planetoid [41] can generate node embeddings based on node features such as character sequence encoding vectors, it can handle the OOV issue and is chosen as a baseline model.

Synonym Discovery. A variety of methods have been proposed to detect synonyms of medical terms, ranging from utilizing lexical patterns [39] and clustering [21] to the distributional semantics models [11]. There are some more recent works on automatic synonym discovery [30, 31, 37, 42]. For example, Wang et al. [37] try to learn better embeddings for terms in medical corpora by incorporating their semantic types and then build a linear classifier to decide whether a pair of medical terms is synonyms or not. Qu et al. [30] combine distributional and pattern based methods for automatic synonym discovery. However, many aforementioned models focus on finding synonyms based on raw texts information, which is not suitable for our privacy-aware clinical data. In addition, nearly all methods could only find synonyms for terms that appear in the training corpus and, thus cannot address the OOV query terms.

6 CONCLUSION

In this paper, we study synonym discovery on privacy-aware clinical data, which is a new yet practical setting and consumes less sensitive information to discover synonyms. We propose a novel and effective framework named SurfCon that considers both the surface form information and the global context information, can handle both InV and OOV query terms, and substantially outperforms various baselines on real-world datasets. As future work, we will extend SurfCon to infer more semantic relationships (besides synonymity) between terms and test it on more real-life datasets.

ACKNOWLEDGMENTS

This research was sponsored in part by the Patient-Centered Outcomes Research Institute Funding ME-2017C1-6413, the Army Research Office under cooperative agreements W911NF-17-1-0412, NSF Grant IIS1815674, and Ohio Supercomputer Center [6]. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S.Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notice herein.

REFERENCES

[1] M. Ballesteros, C. Dyer, and N. A. Smith. 2015. Improved transition-based parsing by modeling characters instead of words with LSTMs. In *EMNLP*.
[2] A. L. Beam, B. Kompa, I. Fried, N. P. Palmer, X. Shi, T. Cai, and I. S. Kohane. 2018. Clinical Concept Embeddings Learned from Massive Sources of Medical Data. *arXiv preprint arXiv:1804.01486* (2018).
[3] O. Bodenreider. 2004. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic acids research* 32, suppl_1 (2004), D267–D270.
[4] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. 2016. Enriching word vectors with subword information. *TACL* (2016).
[5] Z. Cao, T. Qin, T. Liu, M. Tsai, and H. Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *ICML*.
[6] Ohio Supercomputer Center. 1987. Ohio Supercomputer Center. http://osc.edu/ark:/19495/f5s1ph73
[7] D. A. Dorr, W.F. Phillips, S. Phansalkar, S. A. Sims, and J. F. Hurdle. 2006. Assessing the difficulty and time cost of de-identification in clinical narratives. *Methods of information in medicine* (2006).
[8] S. G. Finlayson, P. LePendu, and N. H. Shah. 2014. Building the graph of medicine from millions of clinical narratives. *Scientific data* 1 (2014), 140032.
[9] S. L Garfinkel. 2015. De-identification of personal information. *NISTIR* (2015).
[10] W. H. Gomaa and A. A. Fahmy. 2013. A survey of text similarity approaches. In *IJCA*.
[11] M. Hagiwara, Y. Ogawa, and K. Toyama. 2009. Supervised synonym acquisition using distributional features and syntactic patterns. *IMT* (2009).
[12] W. Hamilton, Z. Ying, and J. Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*.
[13] K. Hashimoto, Y. Tsuruoka, R. Socher, and o. 2017. A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks. In *ACL*.
[14] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush. 2016. Character-Aware Neural Language Models.. In *AAAI*.
[15] D. P. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.
[16] P. LePendu, S. V. Iyer, C. Fairon, and N. H. Shah. 2012. Annotation analysis for testing drug safety signals using unstructured clinical notes. In *Journal of biomedical semantics*, Vol. 3. BioMed Central, S5.
[17] O. Levy and Y. Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *ACL*.
[18] O. Levy and Y. Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *NeurIPS*.
[19] J. Liang, P. Jacobs, J. Sun, and S. Parthasarathy. 2018. Semi-supervised embedding in attributed networks with outliers. In *SDM*.
[20] H. J. Lowe, T. A. Ferris, P. M. Hernandez, and S. C. Weber. 2009. STRIDE–An integrated standards-based translational research informatics platform. In *AMIA*.
[21] Y. Matsuo, T. Sakaki, and K. Uchiyama. 2006. Graph-based word clustering using a web search engine. In *EMNLP*.
[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013. Efficient estimation of word representations in vector space. *arXiv:1301.3781* (2013).
[23] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*.
[24] J. Mueller and A. Thyagarajan. 2016. Siamese Recurrent Architectures for Learning Sentence Similarity.. In *AAAI*.
[25] P. Neculoiu, M. Versteegh, and M. Rotaru. 2016. Learning text similarity with siamese recurrent networks. In *Workshop on Representation Learning for NLP*.
[26] S. V. Pakhomov, G. Finley, R. McEwan, Y. Wang, and G. B. Melton. 2016. Corpus domain effects on distributional semantic modeling of medical terms. *Bioinformatics* 32, 23 (2016), 3635–3644.
[27] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, et al. 2017. Automatic differentiation in PyTorch. In *NIPS-W*.
[28] J. Pennington, R. Socher, and C. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.
[29] B. Perozzi, R. Al-Rfou, and S. Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*.
[30] M. Qu, X. Ren, and J. Han. 2017. Automatic synonym discovery with knowledge bases. In *KDD*.
[31] J. Shen, R. Lv, X. Ren, M. Vanni, B. Sadler, and J. Han. 2019. Mining Entity Synonyms with Efficient Neural Set Generation. In *AAAI*.
[32] A. Stubbs and Ö. Uzuner. 2015. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/UTHealth corpus. *Journal of biomedical informatics* 58 (2015), S20–S29.
[33] C. N. Ta, M. Dumontier, G. Hripcsak, N. P. Tatonetti, and C. Weng. 2018. Columbia Open Health Data, clinical concept prevalence and co-occurrence from electronic health records. *Scientific data* 5 (2018), 180273.
[34] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. 2015. Line: Large-scale information network embedding. In *WWW*.
[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is all you need. In *NeurIPS*.
[36] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. 2018. Graph attention networks. In *ICLR*.
[37] C. Wang, L. Cao, and B. Zhou. 2015. Medical synonym extraction with concept space models. In *IJCAI*.
[38] Q. Wang, B. Wang, and L. Guo. 2015. Knowledge Base Completion Using Embeddings and Rules.. In *IJCAI*.
[39] J. Weeds, D. Weir, and D. McCarthy. 2004. Characterising measures of lexical distributional similarity. In *COLING*.
[40] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. In *EMNLP*.
[41] Z. Yang, W. W. Cohen, and R. Salakhutdinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *ICML*.
[42] C. Zhang, Y. Li, N. Du, W. Fan, and P. S. Yu. 2018. SynonymNet: Multi-context Bilateral Matching for Entity Synonyms. *arXiv preprint arXiv:1901.00056* (2018).