Reproducibility Project Instructions for CS598 DL4H in Spring 2023

# Sathish Rama and Mitchell Mays

## {sbrama2, mm109}@illinois.edu

Group ID: 195

Paper ID: 175

Presentation link: <https://youtu.be/buR9-KsEjNI>

Code link: N/A for Proposal/Draft

# Introduction

This Project Proposal is to reproduce & report findings of the paper SurfCon: Synonym Discovery on Privacy-Aware Clinical Data[8].

### Background

Extracting knowledge from unstructured clinical texts, synonym discovery, is an important task which can benefit many situations. Automatically discovering synonyms (e.g., "c vitamin", "vit c", "ascorbic acid") or misspelled variations (e.g. "viatmin c") can help to expand the query and thereby enhance the retrieval performance to find valuable information such as patient-clinical interactions and disease treatment outcomes. Due to patient privacy and security, it's not possible to get access to raw or de-identified clinical texts thus medical terms and their aggregated co-occurrence counts extracted from raw clinical texts are becoming a popular substitute for raw clinical texts to study EMR data.

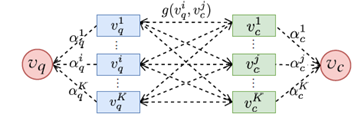
The SurfCon paper proposes a new framework to facilitate synonym discovery, informed by aggregated co-occurrence counts of medical terms extracted from clinical texts..

**Current methods and limitations :** The paper discusses some of the existing methods and how SurfCon helps resolve an unmet need. Existing methods include mapping query terms to a Knowledge Base (KB) to retrieving synonyms, automatic synonym extraction[7] from large corpus of wiki, similarity models between terms (ex: vit c & vitamin c), and semantic similarity models between terms using global context. These methods have limitations such as incompleted or out of date KBs, dissimilar looking synonyms, or a challenge of getting training data in a privacy-aware setting. To address the above limitations, the paper proposes a new method called SurfCon[8].

### New approach proposed in the paper

The use of co-occurrence data as global context combined with the more traditional term embedding comparisons is a new technique for representing and learning relationships between medical terms. The SurfCon[8] framework uses this mechanism to help detect synonyms that are not similar on the surface form and to handle both In-Vocabulary (InV) and Out-of-Vocabulary (OOV) query terms. The architecture consists of the following components:  
 *Fig1: SurfCon Framework*

* Bi-level surface form encoding component: This exploits both character and word-level information to encode a medical term into a vector. It then computes a surface score of two terms based on their encoding vectors. This works well for detecting synonyms similar on surface form only.
* Context matching component: This component utilizes the co-occurrence graph as a representation of a term’s global context. Based on a training set of term’s contexts, it seeks to predict the global context of a novel term. This allows SurfCon to construct a global relational representation of new terms to the existing library of medical terms.

*Fig2: Dynamic context matching mechanism*

Given a query term, the bi-level surface form encoding component and the context matching component rank the likelihood of a candidate term being synonymous based on the surface form information and global context information.

# Scope of reproducibility

### The paper compares the novel SurfCon[8] approach with 10 baseline methods including Surface form only, Global Context only, and hybrid methods. As a part of this comparison, the paper includes variants of the SurfCon architecture (without context, static context matching, and dynamic context matching). For the reproduction study, we have reproduced SurfCon without context matching, and SurfCon with dynamic context matching to verify the claims mentioned in the paper. Specifically, we sought to verify that the SurfCon method improves the baseline methods as indicated in the paper, and to show that each of the novel components included in the SurfCon architecture are additive.

### The experiments produced in the original SurfCon[8] paper included multiple evaluations criteria.. These criteria are noted below. Criteria included in our reproduction study are indicated with a \*.

* \*1 Day Dataset: *Co-occurence graph based on terms co-occuring in a Patient’s notes in the same day*
* All Day Dataset: *Co-occurence graph based on terms co-occuring in a Patient’s notes for all time*
* \*Dev: *Evaluation of Validation Set used for evaluation of model training and hyperparameter tuning*
* \*InV Test: *Test Set of Query Terms included in Co-frequency vocabulary*
* \*InV Dissim: *Portion of InV Test where Synonyms are visually dissimilar from query term*
* OoV Test: *Test Set of Query Terms not-included in Co-frequency vocabulary*
* OoV Dissim: *Portion of OoV Test where Synonyms are visually dissimilar from query term*

### Due to the unavailability of one of the original pre-trained embeddings datasets used in the Surfcon paper, our team introduced an alternative. Rather than CharNGram[3], we utilized the FastText[11] pre-trained subword embeddings which were trained using the Wikipedia corpus. This alternate embedding method provided an additional opportunity to evaluate the significance of the embeddings approach in the full model architecture.

### Addressed claims from the original paper

Claim 1: The SurfCon (Surf-only) architecture improves upon existing methods for Synonym Generation

* InV Test & InV Dissim Metrics are greater than comparators

Claim 2: The full SurfCon architectures further improves the Surf-only model for Synonym Generation

* InV Test & InV Dissim Metrics are greater than Surf-Only

# Methodology

### Model descriptions

The SurfCon architecture consists of two models, the Context Prediction Model and the Term Ranking Model.

The Context Prediction model is composed of pre-trained subword and word embeddings, and a fully connected layer. Based on a query term and list of potential candidate terms, the model concatenates the embeddings and uses the fully connected layer with LogSoftMax to predict the probability for each candidate term to appear in a query term’s context. The model uses an Adam optimizer, and loss is calculated using CrossEntropyLoss between co-frequency graph data and predicted outputs. The core hyperparameters for this model are learning rate and embedding dimensions. While embedding dimensions are not fixed, since this work utilizes pretrained embeddings, the embedding dimension is based on the pretrain word and subword embeddings.

The Term Ranking model consists of a TermEncoder (an encoding model for surface form scoring), the Context Prediction model described above, and LINE[4] node embeddings. Cosine similarity is used to generate the surface score and bilinear similarity for the context score. The node embeddings are used to create feature vectors from context data which are used for the dynamic context matching mechanism. The hyperparameter γ is used to weigh the significance between context score and surface score. Additionally, the top number of predicted contexts to include for each term in the context term can be set as a parameter. This model also uses an Adam optimizer. Loss is cross entropy and utilizes a ranking framework introduced by ListNet[13].

Although the two models described are nested into the Term Ranking model, they are called out separately as training is performed in two phases: first for the Context Predictor model, then for the rest of the Term Ranking model.

### Data descriptions

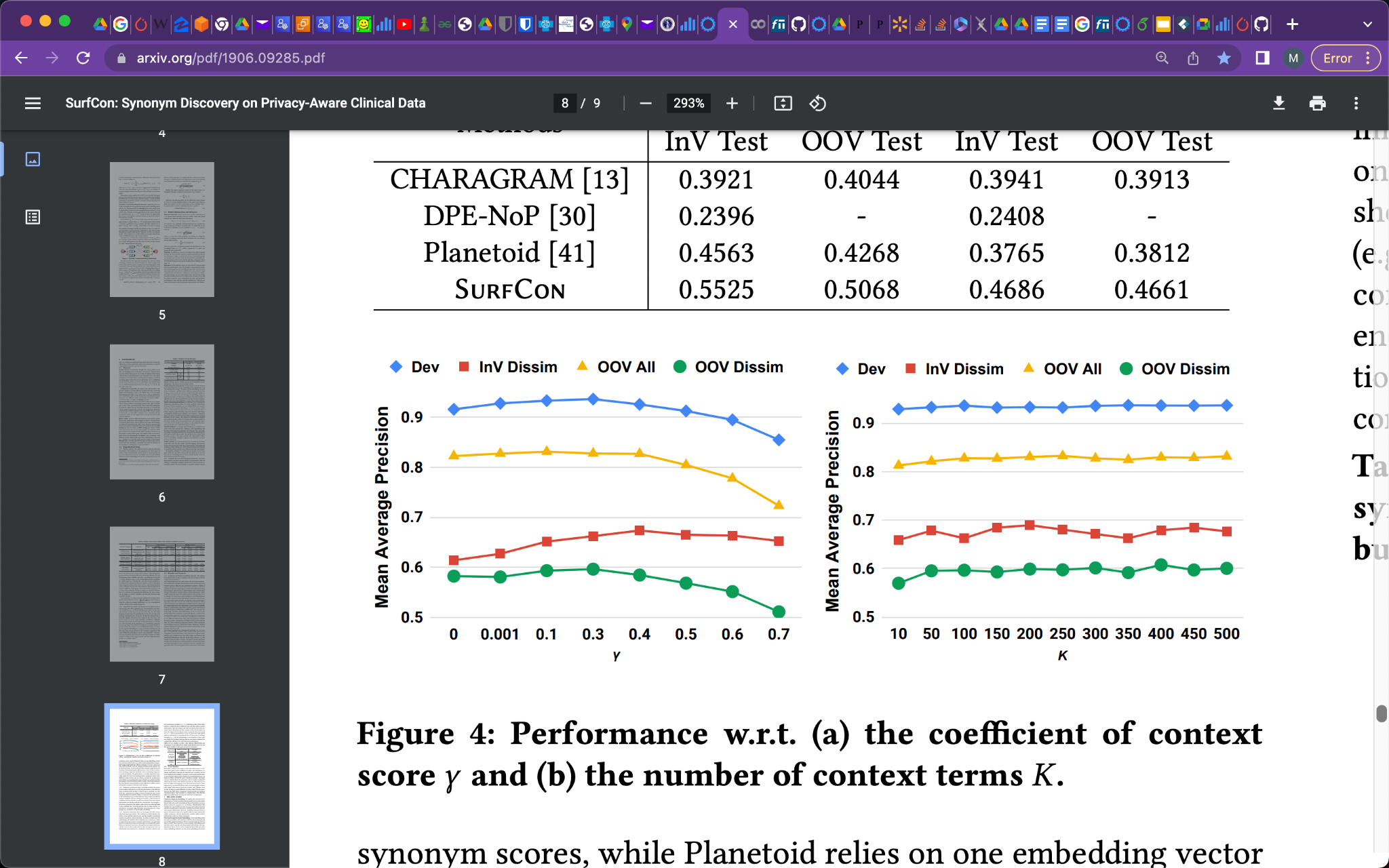
The intent of this work is to utilize a version of clinical data that is free of patient privacy information which is accomplished by utilizing a dataset that contains a co-occurrence frequency graph of terms mapped to standardized medical terms. In addition, a few supplementary datasets are utilized for building the model. As described in the introduction, it is necessary to query the final model with terms that do not reside in the co-occurrence graph in order to perform verification of the model. This necessitates an OOV dataset with mapped synonyms. The dataset is built by sampling UMLS CUI for terms that do not exist in the graph and mapping them to the synonyms that are present in the graph. To test the challenging synonyms that do not share a common surface form, a subset of this OOV dataset is sampled by identifying the most dissimilar synonym.

The last dataset used to enable this work is an existing embedding reference table for character-based embedding. As mentioned in the introduction, a surface form embedding of the terms is done using character n-grams. This embedding is accomplished by making use of a pre-trained character n-gram embeddings from Hashimoto et al. [3]. These embeddings are available in their public Github repository.

For the purpose of reproduction, we have used fasttext embedding instead of the above pre-trained character n-gram as used in the paper.

### Hyperparameters

As the majority of the focus of our reproduction study was on data setup, pre-processing and integrating a new form of subword embeddings, we did not heavily explore variations of model hyperparameters. The original authors thoroughly demonstrated the impact of adjusting the fundamental model hyperparameters (γ and number of contexts).



*Fig3: Gamma and Number of Contexts Hyperparameter impact*

Therefore, based on their insights we utilized the recommended hyperparameter values. The coefficient of context score(γ) balances the surface score and the context score and we have used γ = 0.3. This value gave the highest mean average precision and intuitively gives some weight to the global context while maintaining the primary significance from the surface score. For *K* (number of predicted contexts), we used 50. The authors noted that using fewer than this had higher sensitivity to variation.  
.

### Implementation

### Reproduction

To reproduce SurfCon[8] from the reference paper we have utilized the existing code generated in the reference paper [3] and written additional code needed to generate our embedding, recreate co-occurrence graph, implemented PPMI algorithm, concept to term mappings and string to term mappings. Our forked and updated code is stored here[[1]](#footnote-0).

We cloned the existing code referenced in the paper and setup the environment needed on the Google Colab. Below are the list of tasks/changes we have done for this reproduction :

1. Identified latest versions for all the dependencies and installed them.
2. Downloaded the data and pre-trained models & embeddings.
   1. CharNgram pre-trained char embedding used in the paper is no longer available at the link. We reached out to Authors, but they also didn’t have the embeddings, so we selected a replacement fastext .
   2. Term and Concept mappings as expected in the code were missing in the original code/datasets so we used the raw text files referenced in the paper and created a utils function to generate the mappings needed.
   3. Additional pre-processing discussed in the paper did not have corresponding code. This included:
   * Conversion of Term frequency to PPMI: We implemented the PPMI algorithm and applied it to the raw graph data available in the dataset. This replaced the term cofrequency with the PPMI.
   * Subsampling of common terms using method from reference paper [12]. Based on a reference of the original paper, we implemented the algorithm used for subsampling terms. This algorithm was used to reduce the term list, removing the most frequent terms.

### Computational requirements

The paper doesn’t mention compute requirements so we reached out to authors for the same. Based on our understanding, below is what we expect. We used GPU’s and Google Colab.

**Proposal Estimation:**

**Experiment1 :** SurfCon (Surf only) vs SurfCon

Data Volumes :Term Co-occurrence graph size : 7.1 GB , # of Nodes : 52,804 , # of Edges : 16.2 Million

Memory : 15GB to 32GB

CPU/GPU Type : 1-2 Tesla T4 GPU’s Each Tesla T4 has: # of cores: 2560 Memory : 16 GB

# of GPU Hours : 2-4 Hrs for training ( 2 x 1-2 hrs )

**Experiment2 :** SurfCon with new encodings

Data Volumes : Same as above

Memory : Same as above

CPU/GPU Type : 1-2 Tesla T4 GPU’s Each Tesla T4 has: # of cores: 2560 Memory : 16 GB

# of GPU Hours : Same as above – with additional 2-4 Hrs to train new Encoding method

**Actual Resource Usage:**

Data Volumes : Same as proposal

Memory : 40GB RAM , 20 GB of GPU RAM

CPU/GPU Type : A100 GPU’s ( lower GPU configuration errored out )

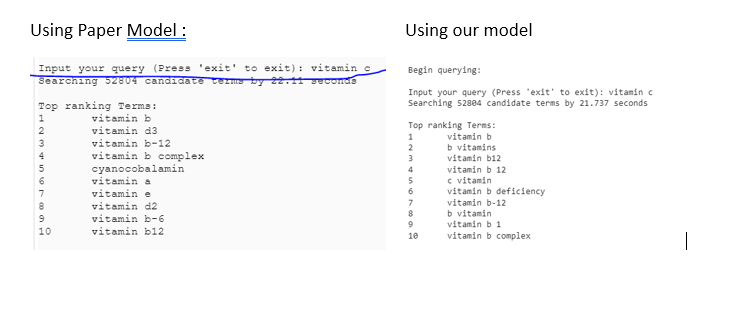
# of Hours : For 20 epochs of training - 3 hours ( with context ) , 1 hour ( without context )

# Results

1. Tested existing code using pre-trained saved models: We were able to execute code using the pre-trained saved models and obtain results.
2. Train the model from scratch

* Term Ranking Model : After creating missing files such as co-occurrence dictionary, term to concept mapping, concept to term mapping, and PPMI values for global counts we initiated the training of the Ranking Model.
* Context Predictor Model : We were successful in training the Context prediction Model.

Below are the side by side results for a sample query term using context.



**Table 1 : Model evaluation in MAP between Paper Claims vs Our reproduction**

| **Methods** | **Paper Claims** | | | **Our reproduction results** | | |
| --- | --- | --- | --- | --- | --- | --- |
| Dev | InV Test(All) | InV Test (Dissim) | Dev | InV Test(All) | InV Test(Dissim) |
| **SurfCon**  **( Surf-Only)** | .9160 | .9053 | .6145 | 0.9465 | 0.9097 | 0.6757 |
| **SurfCon** | .9348 | .9176 | .6821 | 0.921 | .8766 | .6176 |

**Results Analysis and Next Steps :**

We have seen better MAP with our model for without context and slightly lower MAP for our model with context. For training with context, we have used 50 max context terms instead of 100 context terms which could be one reason. Our overall results looks very good and in some cases we have seen our model generating better results.

# Discussion

We are able to reproduce the original paper. Testing the model using the code, provided instructions and the saved checkpoints was easy. Initially, we were not able to train & reproduce the model as is using the codebase since the word embedding used in the paper was not available and preprocessed datasets( co-occurrence graph, ppmi scores, term mappings ) were not included in the original code base. We spent a lot of time and effort to generate all the missing datasets and make the code work on our datasets.

We could recommend the paper authors to provide either the pre-processed datasets (esp. co-occurrence graph with PPMI scores, string to concept mapping, term to concept mapping) or provide a utility function to generate these to help reproducing this easy.

We created & documented this in our notebook to help anyone to enhance this work further.

# Communication with original authors

1. Reached out to inquire about the compute requirements and received response.
2. We realized that charNgram embedding used in the paper is no longer available at the link so discussed with author and decided to use a replacement embedding[11]

References

# 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.

# 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.

# K. Hashimoto, Y. Tsuruoka, R. Socher, and o. 2017. A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks. In ACL.

# J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. 2015. Line: Large-scale information network embedding. In WWW.

# J. Wieting, M. Bansal, K. Gimpel, and K. Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. In EMNLP.

# P. Neculoiu, M. Versteegh, and M. Rotaru. 2016. Learning text similarity with siamese recurrent networks. In Workshop on Representation Learning for NLP.

# M. Qu, X. Ren, and J. Han. 2017. Automatic synonym discovery with knowledge bases. In KDD.

# Zhen Wang , Xiang Yue , Soheil Moosavinasab , Yungui Huang , Simon Lin , Huan Sun 2019. SurfCon: Synonym Discovery on Privacy-Aware Clinical Data, <https://dl.acm.org/doi/pdf/10.1145/3292500.3330894>, KDD ’19, August 4–8, 2019, Anchorage, AK, USA

# W. Hamilton, Z. Ying, and J. Leskovec. 2017. Inductive representation learning on large graphs. In NeurIPS.

# P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. 2018. Graph attention networks. In ICLR.

# T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, A. Joulin. [*Advances in Pre-Training Distributed Word Representations*](https://arxiv.org/abs/1712.09405)

# T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. In NeurIPS

# Z. Cao, T. Qin, T. Liu, M. Tsai, and H. Li. 2007. Learning to rank: from pairwise approach to listwise approach. In ICML.

1. <https://github.com/mitchell-mays/SurfCon_customEncoding> [↑](#footnote-ref-0)