Automated LOINC Standardization Using Pre-trained Large Language Models

Tao Tu TAOTU@GOOGLE.COM
Eric Loreaux ELOREAUX@GOOGLE.COM
Emma Chesley ECHESLEY@GOOGLE.COM
Adam D. Lelkes LELKES@GOOGLE.COM
Paul Gamble PAULGAMBLE@GOOGLE.COM
Mathias Bellaiche MBELLAICHE@GOOGLE.COM
Martin Seneviratne MARTSEN@GOOGLE.COM
Ming-Jun Chen MINGJUNCHEN@GOOGLE.COM
Google Research

Abstract

Harmonization of local source concepts to standard clinical terminologies is a prerequisite for multi-center data aggregation and sharing. Challenges in automating the mapping process stem from the idiosyncratic source encoding schemes adopted by different health systems and the lack of large publicly available training data. In this study, we aim to develop a scalable and generalizable machine learning tool to facilitate standardizing laboratory observations to the Logical Observation Identifiers Names and Codes (LOINC). Specifically, we leverage the contextual embedding from pre-trained T5 models and propose a two-stage fine-tuning strategy based on contrastive learning to enable learning in a few-shot setting without manual feature engineering. Our method utilizes unlabeled general LOINC ontology and data augmentation to achieve high accuracy on retrieving the most relevant LOINC targets when limited amount of labeled data are available. We further show that our model generalizes well to unseen targets. Taken together, our approach shows great potential to reduce manual effort in LOINC standardization and can be easily extended to mapping other terminologies.

Keywords: Large Language Model, T5, LOINC, Contrastive Learning, Sentence Embedding, Data Standardization, Medical Entity Linking

1. Introduction

Electronic health records (EHRs) have become an integral part of the digital healthcare systems in the past decade (Atasoy et al., 2019). Efficient sharing and aggregation of EHRs across various health institutions is essential for improving patient care quality, facilitating public health surveillance, and reducing healthcare costs. EHRs encompass a rich body of clinical information: laboratory tests, clinical observations, physician notes, and medical history. They are often stored in heterogeneous formats and encoded in proprietary schemes specific to an institution (Abhyankar et al., 2012). Such idiosyncrasy in the source encoding has become the major obstacle to the success of multi-site clinical information exchange. Therefore, tools for mapping local clinical data to standard terminologies are crucial for data interoperability across sites, with machine learning-based automation playing a key role.

In this study, we focus on developing machine learning tools to automate the standardization of laboratory observations to the Logical Observation Identifiers Names and Codes (LOINC) (Stram et al., 2020). The LOINC system is a standardized coding system for laboratory observations where each laboratory record is identified across six dimensions: component, property, time, system, scale, and method. LOINC codes can provide fine-grained information useful for resolving the ambiguity often seen in local coding systems. Such ambiguity occurs due to a number of factors: (i) home-grown acronyms and synonyms used by local laboratories; (ii) misspelling and human errors during the manual entry of lab results; (iii) missing information (specimen, unit, instrument, etc.) in the record. As a result, accurately mapping local laboratory observations to LOINC is an onerous, manual task that is resource-consuming and error-prone. There have been efforts to develop tools to automate this data harmonization process (Fidahussein and Vreeman, 2014; Khan et al., 2006; Parr et al., 2018; Kelly et al., 2021). However, this problem remains very challenging because the current LOINC database contains more than distinct 80,000 LOINC codes to choose from, and idiosyncratic local lab identifiers often do not provide sufficient and coherent information to enable an accurate mapping. The majority of existing automated machine learning tools rely heavily on hand-crafted features that are engineered specifically to one data center. Such dependency on complex manual feature engineering significantly limits the scalability and generalization of these tools to other data sources and unseen targets.

In this work, we aim to develop a scalable machine learning tool to automate the mapping from local source codes to target LOINC codes. In particular, we leverage the semantic expressiveness of embeddings from pre-trained large language models (LLMs) and formulate the learning problem in the context of few-shot learning to enable training with a small amount of labeled data (Wang et al., 2020). While it is possible to use our model as a fully automated tool, in practice, clinical personnel is often involved in the loop to ensure accuracy. We propose providing the clinical user with a list of k most relevant suggestions to facilitate the review process. To this end, we focus on the top-k prediction performance metrics in the development and evaluation of our model. Our contributions are as follows: (i) We propose a model that requires minimally manual feature engineering and utilizes only free text information in the source and target codes. (ii) We propose a two-step fine-tuning strategy in combination with data augmentation, which improves model performance over pre-trained LLMs using only a limited amount of source-target pairs or just the target codes alone. The proposed framework is therefore easily adaptable to other standard terminologies even in the absence of source codes. (iii) We employ a contrastive learning approach which enables the model to generalize to unseen target codes without the need of retraining the model during inference.

2. Related work

Previous studies have proposed different approaches to automate LOINC harmonization, considering varying numbers of source codes and LOINC targets of interest. Most of these approaches used either rule-based string matching algorithms or machine learning classifiers. Both types of method rely primarily on hand-crafted features, which makes them difficult to apply on new data sources. For example, Khan et al. (2006); Fidahussein and Vreeman (2014) derived a rich local corpus for LOINC mapping which achieved an accuracy between 63% and 79%.

Sun and Sun (2006) developed a lexical mapping tool that correctly identified 63% local concepts on average. Parr et al. (2018); Kelly et al. (2021) treated LOINC mapping as a multi-class classification problem and trained various classifiers based on a set of hand-crafted textual and numerical features. While both studies showed a relatively high accuracy (ranging from 85% to 95% on various datasets), the major drawback of their approaches is the difficulty to handle unseen LOINC targets as the classifiers were trained to predict only a fixed number (1,164 and 482) of LOINC codes. A more recent study (Langton and Srihasam, 2021) proposed a hybrid approach combining deep learning models and word-logic methods to avoid the need of complex feature engineering. They trained six character-level gated recurrent unit (GRU) classifiers to make prediction on each of the six LOINC dimensions from source text strings and then combined the outputs from six classifiers with logic based method for the final LOINC code selection. Their approach achieved human-level performance of 80% accuracy on 98% of source codes. Our approach, on the other hand, uses embeddings from pre-trained LLM to extract features from text strings, avoiding the need for manual feature engineering. This allows our model to be more scalable and generalizable to different data sources compared to previous approaches. More importantly, since the training data we use only contain one or few source examples for each LOINC target, it makes the classification setting not suitable. Instead, we use a contrastive approach to fine-tune the embeddings from pre-trained LLMs given limited amount of training data. This approach enables the model to generalize to an arbitrary number of LOINC targets at inference stage.

3. Methods

3.1. Datasets

We aggregate source and target pairs from the open-source EHR database MIMIC-III (Medical Information Mart for Intensive Care) (Johnson et al., 2016). In our analysis, we focus on LOINC codes associated with laboratory and clinical observations and utilize only free text information associated with source and target codes. As such, we aggregate all local source concepts in the "d_labitems" table by grouping on the "itemid", "label", "fluid", and "loinc_code" fields. Specifically, for each source code, we concatenate the text terms from the "label" and "fluid" (specimen) fields into a single text string. We then convert all text strings into lower case. This results in 579 source-target pairs with a total of 571 unique LOINC targets, where the majority of these pairs are one to one mapping.

3.2. Data augmentation

We apply data augmentation techniques to create variations in both source and target text strings to overcome data scarcity. We leverage the rich information in the publicly available LOINC table (version 2.72) [1] to augment the training data. In particular, LOINC table provides three variants of text label for each LOINC code: long common name (LCN), display name (DN), and short name (SN). Additionally, for a subset of LOINC codes, the "RELATEDNAMES2" field in the LOINC table provides common acronyms, synonyms, and custom nomenclature related to the code. As a result, we apply character-level random deletion, word-level random swapping, word-level random insertion (of the related names), and word-level acronym substitution to create varia-

1. https://loinc.org/news/loinc-version-2-72-is-now-available/

tions in the text representation of source and target codes. Examples are shown in Figure 1A.

3.3. Contrastive learning

Even after data augmentation, each target class only has a few training examples. Furthermore, our training data only cover a very small percentage of LOINC codes. Motivated by the recent success of constrastive learning in few-shot settings (Chen et al., 2020; Wang et al., 2020; Geng et al., 2019), we propose fine-tuning the embeddings from a pre-trained LLM with a contrastive loss function to learn discriminative latent representations of the textual information in source and target codes, with the goal of reducing within-class variance while increasing between-class separability. Different from a multi-class classification setup, the constrastive approach enables the model to be trained in both supervised and unsupervised settings, which makes the learning possible with target codes alone. Specifically, in order to give our model the capability to handle variants of source/target inputs (acronyms, synonyms, and misspelling), we choose a triplet loss function (Schroff et al., 2015) defined as:

$$L = \max\left(0, \mathrm{D}^2_{f_{\boldsymbol{\theta}}(x_a, x_p)} - \mathrm{D}^2_{f_{\boldsymbol{\theta}}(x_a, x_n)} + \alpha\right)$$

where x_a is an anchor sample, x_p is a positive sample in the same class as x_a, x_n is a negative sample in a different class from the anchor. D represents a distance metric (cosine distance) and $f_{\boldsymbol{\theta}}$ is the trained LLM encoder. α is a margin hyperparameter. Triplet loss function aims at minimizing the distance between the positive sample and the anchor while pushing away the negative sample from the anchor. For example, if we sample a triplet using the long common name of one LOINC code as the anchor, the short name of the same code as the positive sample, and the long common name of a different LOINC code as the negative sample, the loss function then encourages the model to embed the long name and abbreviation of the same code closer in the latent space. Note that both source and target codes can be selected to form a triplet based on their class assignment.

Since previous work (Schroff et al., 2015; Hermans et al., 2017; Robinson et al., 2020; Sikaroudi et al., 2020) has shown that the sampling strategy of selecting triplets significantly impacts the model performance, we employ the online batch-based hard triplets mining with a large batch size to achieve high training efficiency as suggested by Schroff et al. (2015). In particular, we compare the model performance of two mining strategies[2]: hard negative mining and semi-hard negative mining. In each iteration, all possible triplets among samples in a mini-batch are evaluated but only valid triplets contribute to the loss depending on the specific sampling strategy. Batch-wise online triplets mining offers some regularization effect since samples are randomly selected within each mini-batch.

3.4. Model architecture

Our model uses a Text-to-Text Transfer Transformer (T5) encoder as the backbone, which takes the raw text string of source/target codes as input. T5 is a family of encoder-decoder transformer models pre-trained in a multi-task setting. A T5 model can scale up to billions of parameters and achieve state-of-the-art performance in a wide range of NLP tasks (Raffel et al., 2020). As shown in Figure 1B, our model uses only the encoder part of T5 to convert a raw input text string to a 768-dimensional embedding vector. The T5 contextual embedding vector is then projected down to a low-dimensional

2. https://github.com/tensorflow/addons/blob/b2dafcfa74c5de268b8a5c53813bc0b89cadf386/tensorflow_addons/losses/triplet.py

space (D = 128) to obtain the final embedding vector via a fully-connected layer, which is L_2-normalized before feeding into the triplet loss function.

The T5 encoder is initialized with pre-trained Sentence-T5 (ST5) model checkpoints. ST5 is a collection of encoder-only T5-style models pre-trained with a contrastive approach similar to Sentence-BERT/RoBERTa (Reimers and Gurevych, 2019). Ni et al. (2022) showed that ST5 achieves new state-of-the-art performance on sentence transfer tasks and outperforms Sentence-BERT/RoBERTa across multiple semantic textual similarity (STS) tasks which are of similar task type as the LOINC standardization task. In addition, empirical results have shown that scaling up ST5 from millions to billions of parameters produces consistent further improvements (Ni et al., 2022). For these reasons, we choose to fine-tune the ST5 encoder for our task. A variety of model checkpoints with varying model sizes are available on TensorFlow Hub.[3] We choose the ST5-base model which employs a 12-layer transformer architecture.

To fine-tune the model, model weights of the T5 backbone are kept fixed and only parameters of the add-on fully-connected layer are updated. This design decision aims to avoid over-fitting during the fine-tuning as only limited amount of training data are available and also allow the fine-tuning to finish within a reasonable time without consuming excessive computational resources.

3.5. Two-stage fine-tuning strategy

To optimize the model performance with very limited training data, we propose a two-stage fine-tuning strategy:

(1) In the first stage, model fine-tuning is done only with the target codes—all LOINC codes in the LOINC catalog. In particular, we extract a total of 78,209 LOINC codes in the laboratory and clinical categories as an auxiliary training dataset. For the majority of codes, we obtain three variants of their text labels. Prior to training, we apply the data augmentation proposed in Section 3.2 on this LOINC target-only auxiliary dataset. The goal of this stage is to fine-tune the model to distinguish among distinct LOINC targets. Since this target-only auxiliary dataset is relatively large and contains a rich amount of information on the targets, the model will gain contextual knowledge about the LOINC ontology, which could boost the performance on the actual source-to-target mapping task.

(2) In the second stage, we further fine-tune the model on the source-target pairs from MIMIC-III. Similarly, we generate a large number of augmented samples for each source/target code and add dropout layer before the fully-connected layer to mitigate over-fitting. This second stage fine-tuning aims to enable the model to learn the specific data distribution unique to the source-target pairs and to jointly embed source and target codes in the same feature space.

3.6. Training and evaluation

The objective of training is to minimize the triplet loss function. It is different from the task during inference, which is to retrieve the most relevant LOINC candidates for each source code. As this work aims at reducing the manual effort in the data harmonization process, it is important to have the correct target ranked as highly as possible among all targets of interest. Thus, to evaluate model performance, we first computed the embeddings for the test source code and all LOINC targets of interest (571 codes), and then used

3. https://tfhub.dev/google/collections/sentence-t5/1

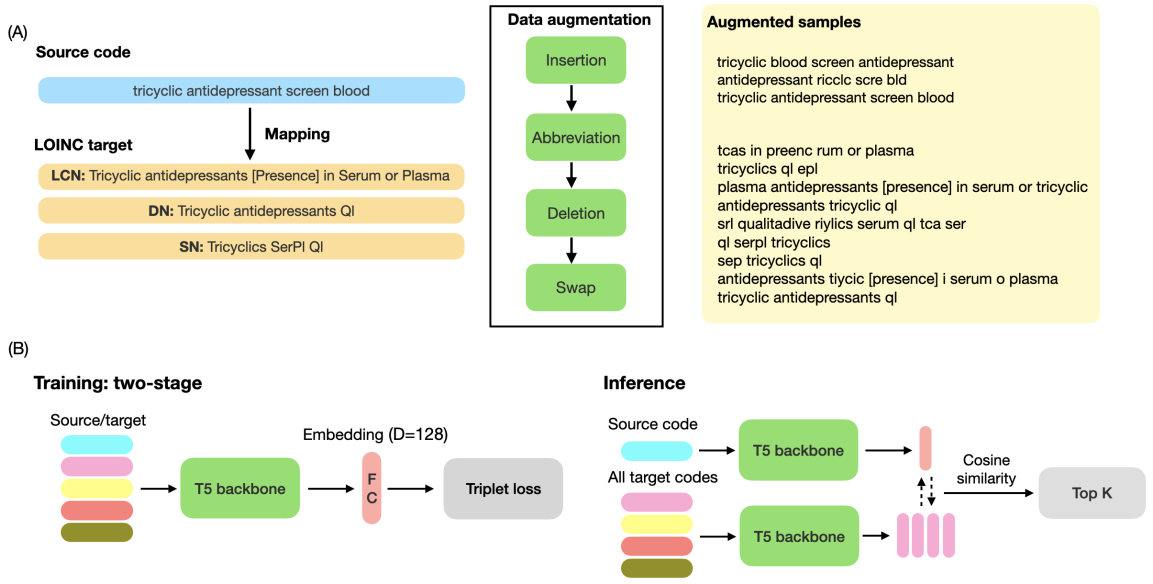

Figure 1: Data augmentation and model architecture. (A) Example of a source code and the corresponding LOINC target. LCN: long common name; DN: display name; SN: short name. A series of data augmentation steps are applied to both source and target strings during training. (B) Diagram of the two-stage fine-tuning and inference. The first stage uses all 78,209 LOINC targets without source codes as the auxiliary training data. The second stage uses a small number of source and LOINC pairs from a specific data source to fine-tune the model.

the cosine similarity to select the top k closest LOINC targets to the source. In particular, we calculated the top-k accuracy as the performance metric, which is defined as the percentage of samples whose correct target is in the top k model predictions.

Fine-tuning experiments were performed for the following purposes: (1) to compare T5 embeddings with other baseline embedding models; (2) to evaluate the gain in performance by the first stage fine-tuning utilizing only LOINC targets; (3) to test the performance improvement from the second stage fine-tuning on a specific dataset with source-target pairs; (4) to assess how well our model generalizes to variations in source representations and unseen targets not used during training. We implemented our model with TensorFlow Abadi et al. (2016) and trained on NVIDIA Tesla V100 GPU with 16 GB memory. All models were trained with Adam optimizer and a batch size of 900. The margin parameter α was set to 0.8 in all experiments.

First stage training and evaluation In this stage, we randomly shuffled the target-only dataset into training (80%) set, validation (20%) set. We trained the model using a learning rate of 1×10^{-4} for 30 epochs to avoid over-fitting. Since no source codes were used in this stage, the performance metrics

were evaluated on all source-target pairs in MIMIC-III.

Second stage training and evaluation In this stage, we trained the model on augmented source-target pairs using 5-fold cross-validation scheme with a smaller learning rate of 1×10^{-5}. Within each fold, 20% of data were held out as test set, we further split the training data into training (80%) set and validation (10%) set for convergence monitoring. We aimed to distinguish between two types of generalizability: performance on the same LOINC targets with different source encoding (Type-1) and performance on unseen new LOINC targets (Type-2). To this end, the performance metrics were computed for the held-out test set before and after data augmentation. The augmentation techniques create random deletion, insertion, and substitution on test samples to mimic the heterogeneity one would observe in real-world applications, which in turn increases the size of the test set approximately 100 times. We reported model performance on the augmented test set in order to assess the robustness of our model against variations in source representations for the same LOINC targets (Type-1). Moreover, to demonstrate the generalizability (Type-2) of our model on unseen targets, we expanded the target set of interest by adding the top 2,000 most common LOINC codes, resulting in a total of 2,313 unique target codes to select from. Note that the additional LOINC targets which are not in MIMIC-III were not included in the second stage fine-tuning process.

4. Results

4.1. Performance of pre-trained LLMs

We first showed off-the-shelf performance of pre-trained language models together with the TF-IDF baseline model. Table 1 shows the performance metrics calculated from different models on MIMIC-III source-target pairs. Interestingly, sentence embeddings computed from BERT (mean pooling of output tokens) and universal sentence encoder (USE) (Cer et al., 2018) perform worse than TF-IDF (frequency-weighted bag-of-words model). This is probably due to the anisotropy phenomenon of contextual embeddings from pre-trained language models, where the direction of vectors in the semantic space is not uniformly distributed (Ethayarajh, 2019; Gao et al., 2021). This phenomenon has been shown to cause the collapse of embeddings that prevents the model from performing well on distance-related metrics (Ni et al., 2022). This problem can be much alleviated by applying contrastive learning to sentence embeddings, which may explain why STSB-BERT/RoBERTa and ST5 yield much better performance. Table 1 presents the performance of two ST5 models of different model size. Overall, model performance increases as the model size increases. ST5-base yields similar performance as STSB-BERT[4] but outperforms STSB-RoBERTa[5] on our task. Notably, using embeddings from pre-trained LLMs yields decent performance even without any fine-tuning.

4.2. Performance of the first stage fine-tuning

Table 2 shows the comparison of model performance after the first stage of fine-tuning using only the target corpus. Training with general LOINC ontology boosts the model performance on the downstream mapping task. As shown in Table 2, fine-tuning with both hard negative and semi-hard negative mining strategies leads to performance in-

4. https://huggingface.co/sentence-transformers/stsb-bert-base
5. https://huggingface.co/sentence-transformers/stsb-roberta-base

Table 1: Performance of different pre-trained language models on MIMIC-III dataset. Only TF-IDF baseline requires training process.

Model	# Parameters	Top-1 accuracy	Top-3 accuracy	Top-5 accuracy
TF-IDF	N/A	58.38%	69.43%	77.03%
USE	256M	25.04%	33.51%	40.93%
BERT	110M	36.78%	48.70%	55.44%
STSB-RoBERTa	110M	50.59%	65.63%	71.50%
STSB-BERT	110M	57.69%	71.68%	76.68%
ST5-base	**110M**	**54.06%**	**71.68%**	**77.72%**
ST5-large	335M	60.00%	81.00%	85.66%

crease over the pre-trained model and TF-IDF baseline performance, especially with respect to the top-1 and top-3 accuracy. Moreover, we observed a significant performance increase after fine-tuning with the semi-hard negative mining strategy. Overall, the semi-hard negative mining strategy seems to perform better than the hard negative mining strategy. We argue this may be because the negative mining strategy focuses on less hard negatives and therefore more negative samples contribute to the loss during training. This may lead to better generalizability of the learned embeddings especially given a large number of target codes used in this stage. These results demonstrate the effectiveness of our contrastive approach in learning discriminative embeddings with only target terminology in the absence of source codes.

4.3. Performance of the second stage fine-tuning

Next, we evaluated the model performance further fine-tuned with source-target pairs. Performance was evaluated using varying sizes of target set. Table 3 presents the performance metrics obtained from 5-fold cross-validation on both the raw test set and the augmented test set. Fine-tuning with both source and target codes further improves the model performance across all measures compared to the starting point performance from the first stage fine-tuning. Overall, the hard negative mining strategy outperforms the semi-hard negative mining strategy probably because the hard negative mining allows the model to focus on the most difficult negatives in a batch and these negatives may have higher impact on model performance in this stage when the training data size is much smaller. When the model was evaluated on the augmented test samples, it shows only small decrease in performance, which indicates the good robustness of our model against the variability in source representations. It is also reasonable that when we expanded the target set to include unseen targets, the model performance decreases as the task gets more difficult. The impact of expanded target set is more prominent on the top-1 accuracy than the top-5 accuracy. When compared to the baseline TF-IDF and pre-trained ST5 models, however, we still observed a gain in performance even after expanding the target set from 571 to 2,313. Taken together, these results suggest that our two-stage fine-tuning strategy enables the model to achieve improved performance not only over pre-trained and baseline models but also across two types of general-

Table 2: Performance improvement after first stage fine-tuning of a ST5-base model using only the target corpus. Performance is evaluated on MIMIC-III source-target pairs whose source codes are not used during training. Results of two online triplets mining strategies are presented.

Method	Top-1 accuracy	Top-3 accuracy	Top-5 accuracy
No training	54.06%	71.68%	77.72%
Hard negative mining	62.35%	77.55%	84.28%
Semi-hard negative mining	**68.05%**	**81.69%**	**89.12%**

izability (variations in source representations and unseen targets). In addition, to support the necessity of our proposed first stage fine-tuning, the model performance of excluding the first stage fine-tuning is also provided in Table 4 where one can see that fine-tuning without the first stage leads to lower model performance compared with fine-tuning with both stages.

5. Discussion and future direction

We propose a contrastive learning framework for LOINC standardization by fine-tuning pre-trained T5 embeddings. Collectively, the experiment results demonstrate that the proposed approach can retrieve the most relevant LOINC targets for local laboratory source codes with a high accuracy. Using a pre-trained ST5 encoder as the feature extractor, our model takes raw free texts as input, which avoids the need for complex manual feature engineering and therefore can be easily generalizable. Since real-world labeled data is often difficult to acquire, we propose a two-stage fine-tuning strategy that leverages the abundance of unlabeled data in the general LOINC ontology. We show that utilizing only the target corpus leads to consistent performance increase in the downstream task. Furthermore, the model fine-tuned on source and target pairs generalizes well in terms of the heterogeneity in source representations and unseen targets. To summarize, our model shows great potential to be deployed as an automated mapping tool to reduce the manual labeling effort and improve the quality of existing mapping. The proposed contrastive representation learning framework provides the model with the flexibility to be easily extended to not only arbitrary number of new targets but also other medical ontologies.

It is noteworthy that our study used a high-quality research dataset whose data distribution may be very different from the real-world datasets, which are mostly proprietary and not accessible to the public. Source codes of the same target often exhibit a high level of heterogeneity in real-world applications. In MIMIC-III, however, most of LONIC targets are paired with only one or few representations of source terms. The small size of labeled data poses a great challenge for training large-scale LLMs. As a result, we chose not to fine-tune the entire LLM but instead we only adjusted the weights of the projection layer during training. We chose pre-trained ST5-base as the backbone of our model due to its good performance and moderate model size. While scaling up the model size shows increased task performance, it also demands much more computing re-

Table 3: Cross-validation performance after the second stage fine-tuning of a ST5-base model using source-target pairs. Two types of generalization performance are evaluated using the augmented test samples and expanded target set. Results are averaged across 5 folds and reported for two online triplets mining strategies, respectively.

Test set without augmentation

Target size	Method	Top-1 accuracy	Top-3 accuracy	Top-5 accuracy
571	Hard	$63.70 \pm 4.83\%$	$81.70 \pm 3.26\%$	$88.26 \pm 3.20\%$
	Semi-hard	$58.03 \pm 7.29\%$	$79.28 \pm 3.21\%$	$85.26 \pm 2.55\%$
2313	Hard	$49.92 \pm 6.06\%$	$73.93 \pm 1.94\%$	$80.84 \pm 3.31\%$
	Semi-hard	$45.43 \pm 7.66\%$	$69.09 \pm 4.55\%$	$78.75 \pm 2.75\%$

Test set with augmentation

Target size	Method	Top-1 accuracy	Top-3 accuracy	Top-5 accuracy
571	Hard	$65.53 \pm 1.85\%$	$81.26 \pm 1.45\%$	$86.52 \pm 1.35\%$
	Semi-hard	$64.62 \pm 1.79\%$	$80.51 \pm 1.26\%$	$86.17 \pm 1.09\%$
2313	Hard	$56.95 \pm 1.49\%$	$73.94 \pm 1.67\%$	$79.98 \pm 1.75\%$
	Semi-hard	$56.38 \pm 1.69\%$	$73.08 \pm 1.35\%$	$79.58 \pm 1.46\%$

Table 4: Model performance with and without the first stage fine-tuning. Results are averaged across 5 folds and reported for the hard negative mining strategy. The first stage fine-tuning using only the target corpus improves the downstream task performance, compared to directly fine-tuning the model on source-target pairs.

Test set with augmentation

Target size	Method	Top-1 accuracy	Top-3 accuracy	Top-5 accuracy
571	stage1+stage2	$65.53 \pm 1.85\%$	$81.26 \pm 1.45\%$	$86.52 \pm 1.35\%$
	stage2 only	$59.81 \pm 1.26\%$	$75.86 \pm 1.13\%$	$81.50 \pm 1.16\%$
2313	stage1+stage2	$56.95 \pm 1.49\%$	$73.94 \pm 1.67\%$	$79.98 \pm 1.75\%$
	stage2 only	$50.89 \pm 1.07\%$	$67.41 \pm 0.93\%$	$73.73 \pm 0.92\%$

sources (infrastructure with TPU support). To further overcome data scarcity, we applied data augmentations designed to create variations in source/target representations that mimic the real-world scenario. The triplet loss function was chosen to encourage these variants of same target/source to cluster together in the latent feature space. We also experimented with two online batch-wise triplet mining strategies together with an offline global triplet mining. Our findings are consistent with Schroff et al. (2015) where semi-hard negative mining performs the best with large data size. However, hard negative mining shows more advantage with small data size in our analysis. Offline mining using global hard negatives is the least efficient and easily leads to over-fitting.

It is arguable that using a language model pre-trained on clinical text, for example, SapBERT model (Liu et al., 2021), might be more beneficial for the LOINC mapping task, compared to the ST5 model which is pre-trained on general domain text. Without any fine-tuning, SapBERT outperforms ST5-base on MIMIC-III, with a top-1 accuracy of 65.98%, a top-3 accuracy of 79.10%, and a top-5 accuracy of 83.25%. However, after the first stage fine-tuning, ST5 outperforms SapBERT after gaining domain-specific knowledge. These results substantiate the effectiveness of fine-tuning directly on LOINC ontology.

There are a number of limitations of this work. First, we did not fine-tune the T5 backbone during training. It is possible that adjusting feature representations of all layers of T5 model will yield further performance gain, especially in the first stage training with the rich target corpus. We will leave this as future work. Second, our model only used raw free text from the source codes as input. We observed that it was difficult for the model to distinguish between the qualitative and quantitative properties of two similar LOINC codes (e.g., "Erythrocytes [#/volume] in Urine by Test strip" vs. "Erythrocytes [Presence] in Urine"). Therefore, future work will explore adding other contextual attributes of the source codes, such as measurement value and unit, to increase the specificity of LOINC mapping. Third, although our model has the capability to generalize to an arbitrary number of targets, we only focused on predicting a subset of targets specific to a dataset in the performance evaluation and experimented with expanding the target set to include the top 2,000 most frequent LOINC codes. While increasing the number of targets improves the LOINC coverage, it also makes the task more difficult and decreases the model performance. The number of targets should be a design choice specific to each application. Lastly, the utility of our model to determine non-mappable source codes (codes with no LOINC target) is not assessed in this study. Since there are a large number of unlabelled source terms in our dataset, addressing this problem will require intensive manual efforts to separate non-mappable codes (true negatives) from codes that haven't been harmonized (false negatives). Moreover, we observed that a small number of source terms lack essential information to allow for accurate mapping. In future work, we will perform comprehensive human-in-the-loop studies including manual validation of the model prediction and human-assigned label to better understand the validity of our model. It is also noteworthy that while we evaluated the model on augmented samples which simulate the real-world data to make our analysis more rigorous, this approach may still lead to over-estimated model performance as the level of heterogeneity in real-world data can be much higher. Nevertheless, even with limited labeled data, our proposed learning framework still demonstrates a great potential for real application use.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

Swapna Abhyankar, Dina Demner-Fushman, and Clement J McDonald. Standardizing clinical laboratory data for secondary use. *Journal of biomedical informatics*, 45(4): 642–650, 2012.

Hilal Atasoy, Brad N. Greenwood, and Jeffrey Scott McCullough. The digitization of patient care: A review of the effects of electronic health records on health care quality and utilization. *Annual Review of Public Health*, 40(1):487–500, 2019. doi: 10.1146/annurev-publhealth-040218-044206.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, 2019.

Mustafa Fidahussein and Daniel J Vreeman. A corpus-based approach for automated loinc mapping. *Journal of the American Medical Informatics Association*, 21 (1):64–72, 2014.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, 2021.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. Induction networks for few-shot text classification. *arXiv preprint arXiv:1902.10482*, 2019.

Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

Jonathan Kelly, Chen Wang, Jianyi Zhang, Spandan Das, Anna Ren, and Pradnya Warnekar. Automated mapping of real-world oncology laboratory data to loinc. In *AMIA Annual Symposium Proceedings*, volume 2021, page 611. American Medical Informatics Association, 2021.

Agha N Khan, Stanley P Griffith, Catherine Moore, Dorothy Russell, Arnulfo C Rosario Jr, and Jeanne Bertolli. Standardizing laboratory data by mapping to loinc. *Journal of the American Medical Informatics Association*, 13(3):353–355, 2006.

John Langton and Krishna Srihasam. Applied medical code mapping with character-based deep learning models and word-based logic. In *Proceedings of the 1st and 2nd Workshops on Natural Logic Meets Machine Learning (NALOMA)*, pages 7–11, 2021.

Fangyu Liu, Ehsan Shareghi, Zaiqiao Meng, Marco Basaldella, and Nigel Collier. Self-alignment pretraining for biomedical entity representations. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4228–4238, 2021.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, 2022.

Sharidan K Parr, Matthew S Shotwell, Alvin D Jeffery, Thomas A Lasko, and Michael E Matheny. Automated mapping of laboratory tests to loinc codes using noisy labels in a national electronic health record system database. *Journal of the American Medical Informatics Association*, 25(10):1292–1300, 2018.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, 2019.

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. *arXiv preprint arXiv:2010.04592*, 2020.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

Milad Sikaroudi, Benyamin Ghojogh, Amir Safarpoor, Fakhri Karray, Mark Crowley, and Hamid R Tizhoosh. Offline versus online triplet mining based on extreme distances of histopathology patches. In *International Symposium on Visual Computing*, pages 333–345. Springer, 2020.

Michelle Stram, Tony Gigliotti, Douglas Hartman, Andrea Pitkus, Stanley M Huff, Michael Riben, Walter H Henricks, Navid Farahani, and Liron Pantanowitz. Logical observation identifiers names and codes for laboratorians: potential solutions and challenges for interoperability. *Archives of pathology & laboratory medicine*, 144(2): 229–239, 2020.

Jennifer Y Sun and Yao Sun. A system for automated lexical mapping. *Journal of the American Medical Informatics Association*, 13(3):334–343, 2006.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020.