CoSiNES: Contrastive Siamese Network for Entity Standardization

Jiaqing Yuan[1], **Michele Merler**[2], **Mihir Choudhury**[2],
Raju Pavuluri[2], **Munindar P. Singh**[1], **Maja Vukovic**[2]
[1] North Carolina State University, Raleigh, NC, USA
[2] IBM Research AI, Yorktown Heights, NY, USA
`{jyuan23, mpsingh}@ncsu.edu`
`{mimerler, choudhury, pavuluri, maja}@us.ibm.com`

Abstract

Entity standardization maps noisy mentions from free-form text to standard entities in a knowledge base. The unique challenge of this task relative to other entity-related tasks is the lack of surrounding context and numerous variations in the surface form of the mentions, especially when it comes to generalization across domains where labeled data is scarce. Previous research mostly focuses on developing models either heavily relying on context, or dedicated solely to a specific domain. In contrast, we propose CoSiNES, a generic and adaptable framework with Contrastive Siamese Network for Entity Standardization that effectively adapts a pretrained language model to capture the syntax and semantics of the entities in a new domain.

We construct a new dataset in the technology domain, which contains 640 technical stack entities and 6,412 mentions collected from industrial content management systems. We demonstrate that CoSiNES yields higher accuracy and faster runtime than baselines derived from leading methods in this domain. CoSiNES also achieves competitive performance in four standard datasets from the chemistry, medicine, and biomedical domains, demonstrating its cross-domain applicability.

Code and data is available at `https://github.com/konveyor/tackle-container-advisor/tree/main/entity_standardizer/cosines`

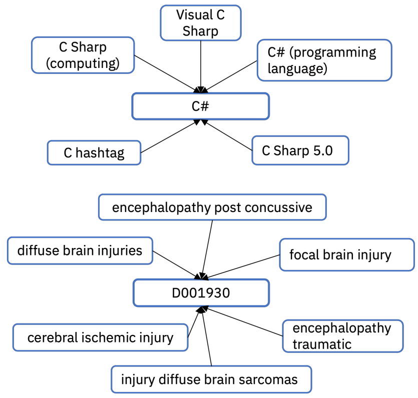

Figure 1: Examples of various mentions referring to the same entity from two different domains. Top: technology, bottom: medical.

1 Introduction

The automatic resolution of mentions in free-form text to entities in a structured knowledge base is an important task for understanding and organizing text. Two well-recognized tasks tackle entity mentions in text. *Entity matching* concerns resolving data instances that refer to the same real-world entity (Li et al., 2020). The data instances usually comprise a specific schema of attributes, such as product specifications. *Entity linking*, also known as entity disambiguation, associates ambiguous mentions from text with entities in a knowledge base, where precise attributes and relationships between entities are curated (Alam et al., 2022). Both tasks involve rich context surrounding the mention and the underlying entity (Li et al., 2020; Alam et al., 2022). Much effort in deep learning approaches focuses on ways to leverage and encode the context surrounding mentions in text and attributes associated with entities in the knowledge base. However, little work has been done on scenarios where such rich context and precise information are not available. In domains such as finance, biology, medicine, and technology, mentions involve specialized jargon, where no context is associated with the mentions and often no attribute of the entities is available other than the mentions themselves.

We tackle the challenge of missing context for entity standardization (ES) mapping, which involves mapping mentions to entities in the knowledge base across multiple domains. Due to the lack of a public dataset for ES and to foster research on the problem, we manually construct a dataset in the technology domain geared to application modernization. We propose an approach called CoSiNES

Proceedings of the First Workshop on Matching From Unstructured and Structured Data (MATCHING 2023), pages 109–119
July 13, 2023 ©2023 Association for Computational Linguistics

for the dataset and then evaluate the generalization of CoSiNES in the biomedical domain.

Application modernization consists in migrating legacy applications to the cloud. It relies on a faithful assessment of the technical components of such applications. Much technical information is contained in free-form textual application descriptions, but automatic extraction of such knowledge is non-trivial due to variations in how the same entities are mentioned (Kalia et al., 2021).

Compared to the two aforementioned tasks of entity matching and linking, ES presents unique challenges. First, the mentions could have acronyms, numbers, symbols, alias, punctuation, and misspellings. Figure 1 shows two examples of multiple mentions referring to the same entity. Second, there is a lack of context surrounding the mentions, and there are no attributes or relationships for entities in the knowledge base, which the previous approaches heavily rely on. Third, large deep learning models require massive training datasets, which are not available for specialized domains. Therefore, architectures that are suited for zero-shot or few-shot learning are of great value for this task.

Another challenge is how to perform entity standardization at scale. A naive way is to have exhaustive comparisons between each possible mention and entity pair, which is inefficient. Previous deep learning models for entity matching and entity linking usually have multiple stages (Papadakis et al., 2020): first stage, such as blocking in entity matching, reduces the number of comparison pairs via a coarse-grained criterion so that the latter stages can focus on filtered candidate pairs. This multistage approach leads to globally inferior performance due to the errors accumulated along the pipeline.

We tackle these challenges with a generic framework based on Contrastive Siamese Network which efficiently adapts domain-agnostic pretrained language models (PLMs) to specific domains using a limited number of labeled examples. Language models have shown great capacity to capture both syntactic and semantic variations of text. Our framework decouples the comparison of mention-entity pairs for training and inference so that the model can be used as a standalone encoder after training. Therefore, the embeddings of the entity from the knowledge base can be precomputed and hashed. At inference time, the running time is linear in the size of query mentions, and we can leverage existing tools, such as FAISS,[1] for efficient and large-scale similarity search.

Our contributions are the following.

- A generic, scalable, and adaptable framework that leverages domain-agnostic pretrained language models.
- A method for generating anchored contrastive groups and a training scheme with a hybrid of batch-all and batch-hard online triplet mining.
- A dataset curated for application modernization, where various mentions for technical components are manually labeled.

We validate these contributions via comprehensive experiments with various hyperparameters, loss functions, and training schemes and show the robustness and effectiveness of the framework on our custom dataset in the technology domain. With optimal settings on our dataset, we further evaluate the framework on four datasets from the biomedical domain. We show that the framework can be adapted to other domains with minimal changes.

2 Related Work

Various forms of entity-related tasks have been studied by previous research, of which three are most relevant to our task.

Entity Matching (EM) identifies if different mentions refer to the same real-world entity, and is an important step in data cleaning and integration (Christen, 2012). The targets of EM are records from a database, where records follow a specific schema of attributes. The goal is to find pairs of records from two databases that refer to the same entity. Whereas early approaches of EM mostly apply rule-based heuristics, recent research often relies on deep neural network (Nie et al., 2019; Mudgal et al., 2018; Li et al., 2020; Ebraheem et al., 2018). As the number of pairwise comparisons grows quadratically, a preprocessing step (blocking) is usually applied to reduce the number of candidate matches. The matcher then takes a pair of a mention and an entity as input and produces a probability of a match. In contrast, entity standardization comes with a predefined set of standard entities, and the mentions come with no attributes. Our method involves learning a metric function, where the model can be used as an encoder to embed mentions and entities in the same space.

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

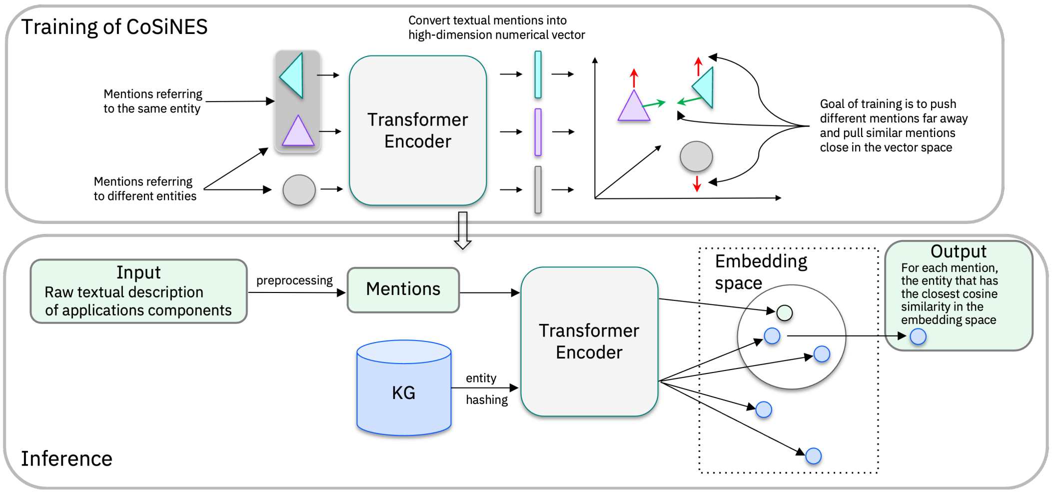

Figure 2: System overview of CoSiNES.

Entity Linking (EL) is the process of linking a mention in context with an entity in a knowledge base. Unlike entity standardization, the entities in the knowledge base, such as WikiData (Vrandečić and Krötzsch, 2014) and Freebase (Bollacker et al., 2008), usually have well-structured attributes and precisely defined relationships between them. The mention comes with rich context and unstructured raw text. To leverage these two different types of contextual information, separate context-mention and graph-entity encoders are designed to produce embeddings respectively, and another neural network is used to combine and project these two embeddings to the same space (Shahbazi et al., 2019; Yamada et al., 2022; Radhakrishnan et al., 2018). Due to the lack of context for both the mention and entity for entity standardization, we propose to use a single unified model as the encoder, which can reduce the complexity of the pipeline.

Entity Normalization (EN) is widely used in the biomedical domain. The task is to map noisy mentions to entities in a well-defined reference set, such as ontologies and taxonomies (Ferré et al., 2020; Ferré et al., 2020). The mentions usually have no context, and the entities come with no attributes, but there is a hierarchical structure in the reference set. Unlike entity standardization in the technology domain, the variations of mentions in life science are fairly standardized and synonyms are rare. The task can be well addressed with a sufficient number of training examples for each entity category, which is not the case in our setting. Fakhraei et al. (2020) propose a similar idea using a Siamese neural network for EN. Our approach differs in the following aspects: the designed training batch-generation algorithm, the computation of the contrastive loss, and the usage of PLMs in our specialized training scheme.

3 Methodology

3.1 Problem Formulation

We denote the set of query mentions as $\mathcal{Q} \equiv \{m_q\}$, and the set of standard entities as $\mathcal{S} \equiv \{e_s\}$. Each entity in $\mathcal{S}$ is associated with zero or more mentions referring to it $e_s \leftarrow \{m_s\}$. Importantly, there should be no overlap between the query mention set $\mathcal{Q}$ and the mentions associated with the standard entity set $\mathcal{S}$. The task is to retrieve an entity $e \in \mathcal{S}$ given $m \in \mathcal{Q}$ such that e is the entity m refers to.

We tackle this task with contrastive learning by learning an embedding encoder such that mentions and entities are encoded to the same high-dimensional embedding space. The property of the embedding space is that the cosine distance between mentions of the same entity is smaller than mentions of different entities.

We design a BERT-based Siamese neural network architecture, which acts as the embedding encoder after training. The training is conducted with a hybrid of batch-all and batch-hard online triplet mining schemes. Figure 2 gives an overview of CoSiNES. The training (top) phase has the goal of pulling similar mentions together and pushing dissimilar mentions far away in the embedding space. After training, the inference (bottom) phase has the goal of using a Siamese neural network to project entities in the knowledge base and query mentions to the same embedding space. At inference time, nearest neighbor search algorithms can be used to retrieve the target entity.

3.2 Contrastive Learning and Triplet Loss

Contrastive Learning (Khan et al., 2022; Rethmeier and Augenstein, 2022; Smith and Eisner, 2005) aims to group similar data points together and push dissimilar data points far apart in a high-dimensional embedding space. Equation 1 shows the core idea of contrastive learning. Here x represents any data point in the domain, x^+ is a positive sample that is similar to x (or from the same class as x), and x^- is a negative sample that is dissimilar to x. E is an encoder, which could be any neural network. And, dis is a distance measure between the embedding vectors.

$$\text{dis}(E(x), E(x^+)) \ll \text{dis}(E(x), E(x^-)) \quad (1)$$

As shown in Equation 2, triplet loss is calculated based on triplets $\{x, x^+, x^-\}$, which consist of two samples from the same class and a third sample from a different class. The intuition is that the distance $\text{d}(x, x^-)$ should be larger than the distance $\text{d}(x, x^+)$ by a *margin*. The *margin* is a hyperparameter that needs to be tuned.

$$\mathcal{L} = \max(\text{d}(x, x^+) - \text{d}(x, x^-) + \text{margin}, 0) \quad (2)$$

Based on the difference between $\text{d}(x, x^-)$ and $\text{d}(x, x^+)$, we can classify triplets into three categories: easy, semihard, and hard. See appendix B for detailed definitions.

3.3 Online Triplet Mining

There are two different strategies of mining triplets for contrastive learning. *Offline mining* generates triplets at the beginning of training. The embeddings of the whole training dataset are computed, then hard and semihard triplets are mined based on the embeddings. Offline mining is highly inefficient. First, it requires computing the embeddings for all the training data to mine the triplets. Second, as the model starts to learn, the hard and semihard triplets may turn into easy triplets. Therefore, at least for a few epochs, we need to update the triplet set frequently. *Online triplet mining* (Schroff et al., 2015) seeks to generate triplets on the fly within a batch. There are two strategies to mine triplets from a batch, i.e., batch all and batch hard. We adopt the same idea in our model and propose a hybrid online mining scheme which is shown to be superior to single-mining strategy.

3.3.1 Batch–All

To form valid triplets, a batch of training data should always include samples from more than one class, and each class should contain at least two samples. Suppose the size of the batch is B and the number of all possible triplets is B^3. However, not all of these triplets are valid as we need to make sure each triplet comprises two distinct samples from the same class and one sample from another class. For all valid triplets in the batch, we simply select all hard and semihard triplets and compute the average loss over them. We do not include easy triplets in computing the average as it will make the loss too small. The calculations are based on the embeddings of the batch after they pass through the model.

3.3.2 Batch–Hard

This strategy always selects the hardest positive and negative for each anchor in the batch. Each data instance in the batch can be used as an anchor. Therefore, the number of triplets is always equal to the size of the batch. The hardest positive has the largest $\text{d}(x, x^+)$ among all positives, and the hardest negative has the smallest $\text{d}(x, x^-)$ among all negatives.

3.3.3 Contrastive Group Generation

Based on the above discussion, a batch should include multiple samples from multiple classes. We sample batches with two steps. First, we randomly generate groups of samples from the same class with size g, and second, we randomly sample b classes of groups to form a batch. Therefore, the effective batch size would be $B = g * b$.

3.4 BERT-Based Siamese Neural Network

The canonical Siamese neural network is an architecture that consists of two towers with shared weights working in parallel on two different inputs. The outputs are passed on to a distance function to learn comparable output vectors. We extend the same idea to a batch of inputs instead of a pair of inputs. We sample the batch as described in Section 3.3 and feed the sampled triplets through the network. The output embeddings of the batch are used to generate valid triplets and compute the loss. The backbone of the Siamese model could be any neural network. We use the pretrained language model BERT (Devlin et al., 2019) as the backbone.

3.5 Hashing and Retrieval

Once the Siamese model is trained, it can be used as a standalone encoder to compute the embeddings of entities and mentions. We precompute the embeddings for all entities and save them for comparisons at inference time. For each query mention, we use the same Siamese model to get the embedding and our task is to retrieve the entity with the closest distance to the mention in the embedding space. For a query set of size q, we need to run the Siamese model only q times, avoiding exhaustive pairwise running of the Siamese model. Potentially, we still need to conduct a pairwise nearest neighbor search over the mention and entity embeddings. Tools such as FAISS can be leveraged to efficiently perform large-scale nearest neighbor search.

4 Experimental Setup

4.1 Dataset

We curate a dataset (ESAppMod) on application modernization that comprises named entities with respect to the technical stack of business applications. There are a total number of 640 unique entities, covering a variety of technical component categories, such as Operating System (OS), Application Server, Programming Language, Library, and Runtime. We manually extract and label 6,412 unique mentions associated with the entities in AppMod from real application descriptions. All annotations are done by domain experts. We split the mentions 60–40 into train and test sets, which yields 3,973 and 2,439 mentions in the training and testing splits, respectively. The mentions associated with each entity are not evenly distributed, ranging from one to over a hundred.

4.2 Hyperparameter Tuning

Implementing our framework involves many design choices and hyperparameters. To facilitate performance at scale, the tradeoff between accuracy and inference time is crucial. We experimented with different sizes of BERT as the backbone of CoSiNES, including BERT-tiny, BERT-mini, BERT-small, BERT-medium, and BERT-base. For triplet mining, we evaluated batch–all, batch–hard, and a hybrid of the two. For the measure of distance, we investigated cosine, Euclidean, and squared Euclidean distance. For the hyperparameters, we evaluated different values of margin, learning rate, and batch size detailed in appendix C. All training experiments were carried out on an NVIDIA A100 GPU with 40GB memory. We use the tool Ray.tune[2] for hyperparameter tuning. Inference times were computed as the cumulative time to predict all 2,439 mentions in the test set on the CPU of Macbook pro with 2.3 GHz Quad-Core Intel Core i7, 32 GB 3733 MHz LPDDR4X RAM. We report the median inference time of 10 runs.

Model	T@1	T@3	T@5	Inf. Time
TF-IDF	69.94	85.36	88.44	60
GNN	67.20	79.29	82.49	29
BERT	32.64	47.23	54.82	17
GPT3	77.24	**90.24**	**93.56**	240
CoSiNES	**80.40**	88.68	90.98	11

Table 1: Experimental results on ESAppMod. T@1: top-1 retrieval accuracy. Inf. Time refers to total inference time in seconds.

4.3 Baselines

We compare CoSiNES with four baselines.

TF-IDF A model that computes TF-IDF embeddings learned from training data(Kalia et al., 2021).

GNN A graph neural network that treats each entity or mention as a chain. Each character represents a node in the graph and its embedding representation is learned during training. The average of the character embeddings are used to represent entity names and mentions (Fan et al., 2022).

BERT We use the mean of last layer outputs of all tokens from BERT_small (Bhargava et al., 2021) to represent entities and mentions. This is the same backbone used to train CoSiNES.

GPT3[3] We use the embedding GPT-3 api from OpenAI to compute the embeddings using model `embedding-ada-002`.

5 Results and Discussions

Table 1 shows the comparative results on our dataset. Our model outperforms all baselines by a significant margin in terms of top–1 retrieval accuracy: 10.46% over TF-IDF, 13.2% over GNN, 47.76% over BERT, and 3.16% over GPT3. Through comprehensive experimentation, we observe that the best performance model has the

[2]https://docs.ray.io/en/latest/tune/index.html

[3]https://beta.openai.com/docs/guides/embeddings/

BERT-small as the backbone. The learning rate is set to 1e−5, contrastive group size is 10, and the batch size of groups is 16, which makes the effective batch size 160. We set the margin to 2.

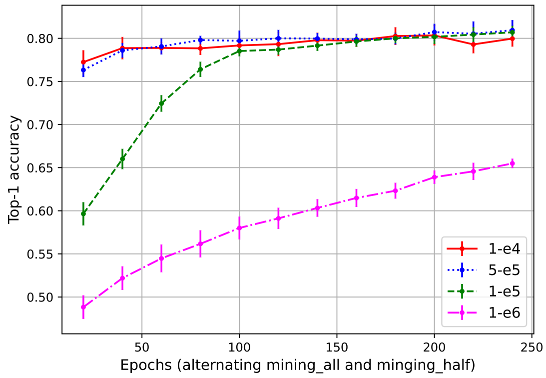

Figure 3: Five–fold cross–validation with different learning rates on training data.

5.1 Learning Rate

To investigate how different learning rates affect the convergence of the Siamese model on our dataset, we run five-fold cross-validation with four learning rates (1e−4, 5e−5, 1e−5, and 1e−6) on the training data, as shown in Figure 3. For each learning rate, we experiment with different numbers of epochs, ranging from 10 to 200 with an interval of 10. The X axis is the number of epochs for each experiment and the Y axis is the top–1 accuracy. The average of the five-fold top–1 accuracy is shown for each dot in the figure, together with the standard deviation across five folds. As we can see, the learning rate affects how fast and stably the model converges, and most of them reach similar performance when trained for enough number of epochs. This indicates that the Siamese model is robust with respect to the learning rate. We set the learning rate to be 1e-5 as it tends to have a smaller deviation of performance.

5.2 Hybrid Triplet Mining

We propose a hybrid of batch–all and batch–hard triplet mining during training. Figure 4 shows the training process with 200 epochs with the above three learning rates, of which the first 100 epochs apply batch–all triplet sampling and the second 100 epochs employ batch–hard triplet sampling. The result shows that for the first batch–all 100 epochs, the training of 1e−4 and 5e−5 is unstable and performance oscillates greatly. When batch–hard mining comes into play, the training becomes much smoother and the performance continues to improve steadily for all three learning rates. This experiment shows that the hybrid mining scheme improves the top–1 accuracy by around 2% compared to the single-mining strategy.

5.3 Model Size

Normally, there is a tradeoff between model accuracy and efficiency. Therefore, we experiment with different sizes of BERT as backbone to find a balance between performance and running time. Figure 5 shows the inference time on the testing set with top–1 accuracy. The results show that CoSiNES with BERT-small achieves the best performance and fast inference time. Although the GPT3 embeddings achieve performance close to CoSiNES, running inference using the GPT3 OpenAI api is inefficient.

5.4 ROC Curve

For a comprehensive comparison between our model and the baselines, we conduct an experiment to compute the receiver operating characteristic (ROC) curve. We add 420 previously unseen relevant but negative mentions from the technology domain that do not refer to any entities in the training set, and calculate the false positive rate under different thresholds. Figure 6 shows that our proposed model has a larger area under the curve, which demonstrates its superior performance over the baselines.

5.5 Qualitative Error Analysis

We examine the predictions from CoSiNES on ESAppMod and categorize the following error types. Table 2 shows a few examples for each of these types.

Misspelling. When a mention has an error in the spelling, the tokens returned by PLMs could be very different, which leads to mismatch. This is a challenge for PLMs, whereas human could easily handle, e.g. “Andriod" vs “Android".

Acronym. Linking acronyms to full expressions seem to be a trivial task for humans, however, CoSiNES falls short of this capability. The rescue might be to design a task specialized for recognizing acronyms for PLMs.

Multi-match. This is the most common error where multiple entities partially match with the mention in the surface form. One way to address this issue is to enrich the training dataset with various mentions, which is not always within easy

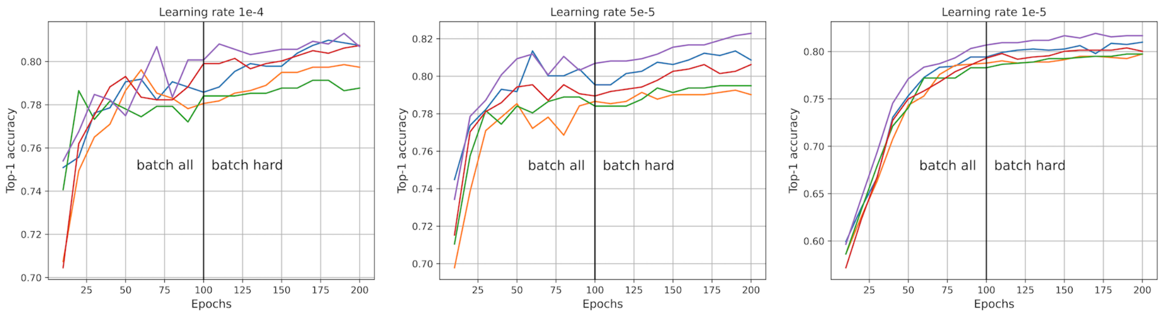

Figure 4: Hybrid triplet mining with different learning rates for five-fold cross validation.

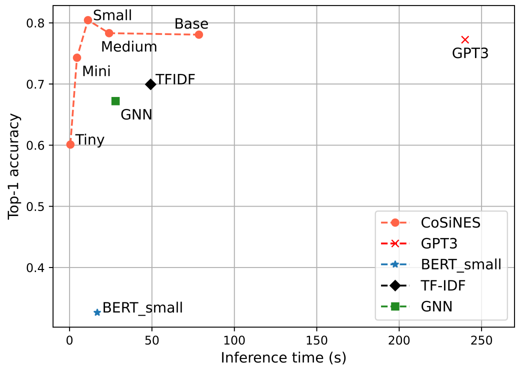

Figure 5: Accuracy versus efficiency between the proposed models on the ESAppMod dataset. The CoSiNES line represents different size of BERT as backbone.

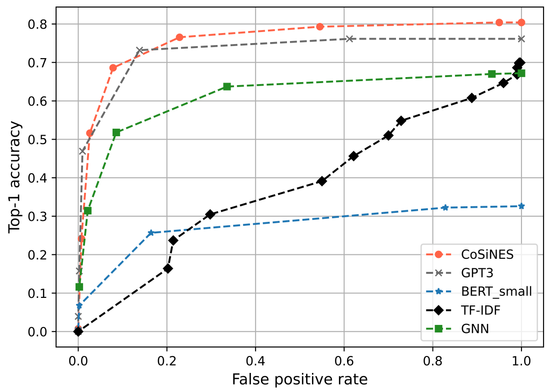

Figure 6: ROC Curves on the ESAppMod dataset.

reach. Another potential approach is to integrate external knowledge about entities so that the model can refer to.

No-match. When the entity and mention have no match at all in the surface form, it is unlikely for the model to retrieve the correct target, especially no context can be leveraged. Therefore, external knowledge could be particularly useful in this case.

6 Adaptation to Biomedical Domain

We show how to adapt our framework to the biomedical domain with minimal changes.

6.1 Datasets

We consider four public datasets, ncbi, bc5cdr-disease, bc5cdr-chemical, and bc2gm, covering three types of entities: chemicals, diseases, and genes. Details and statistics regarding the datasets can be found in apprendix A.

6.2 Baselines

We compare our framework with three models.

TF-IDF Like the baseline for ESAppMod, we implement a straightforward TF-IDF model (Kalia et al., 2021) based on the knowledge database for each dataset and apply nearest-neighbor search for testing.

BioBERT ranking Use BioBERT (Lee et al., 2019) to encode concepts and mentions without fine-tuning. BioBERT is a large biomedical language representation model pretrained with PubMed abstracts and PMC full-text articles.

BioSyn BioSyn (Sung et al., 2020) is the state-of-the-art model for biomedical entity normalization with synonym marginalization and iterative candidate retrieval. The model leverages sparse embedding from TF-IDF and dense embedding from BioBERT.

6.3 Domain Adaptation

For domain adaptation, it would be ideal if we can make none or a few changes to the model architecture and training process. Therefore, we follow all experimental settings, such as learning rate, margin, contrastive group generation, and hybrid training scheme from the experiments on our proposed datasets. The most significant change is that to adapt to a new domain, we use dmis-lab/biobert-

Error type	Mention	Target entity	Top-5 retrieved entities
Misspelling	Andriod Visusal Basic	Android Visual Basic	IBM ILOG Views / Oracle Real-Time Decisions (RTD) / BeOS / Ingres / etcd Clarify|Clear Basic / BASIC / IBM Basic Assembly Language / Pervasive PSQL / ADABAS
Acronym	NES IIB	Netscape Enterprise Server IBM Integration Bus	Mobile / SAS / iOS / Powershell / MinIO Visual Basic / VB.NET / Clarify|Clear Basic / IIS|* / Ada
Multi-match	Cordova Android MQ 9.1 Open Liberty	Apache Cordova IBM Websphere MQ WebSphere Liberty	Android / Apache Cordova / Cisco IOS / Perl|Oraperl / Keycloak Microsoft MQ / MQ Client / IBM Websphere MQ / Qiskit / IBM WebSphere MQ Telemetry OpenROAD / WebSphere Liberty / Virtual Appliance / OpenVPN / Microsoft System Center Endpoint Protection
No-match	AS400 EAP	IBM Power Systems JBoss	DB400 / Asterisk / Primavera P6 / EAServer / Microsoft Excel XAMPP / F5 Secure Web Gateway Services / Java|Java Web Start / UltiDev Web Server Pro (UWS) / A-Auto Job Scheduling Software

Table 2: Examples for each type of errors on ESAppMod.

v1.1[4] in replacement of the regular BERT as our backbone. We conduct all experiments on two NVIDIA A100 GPUs and adjust the batch size for each dataset based on the lengths of the mentions.

6.4 Results

The results are shown in Table 3. We reproduce the BioBERT experiment reported by (Tutubalina et al., 2020a) using the embedding of the [CLS] token as the representation. The results are almost identical. The minor differences might be due to different versions of the pretrained language model.

The performance of BioSyn reported by Sung et al. (2020) is high. However, as pointed out by Tutubalina et al. (2020a), the original testing splits used by Sung et al. (2020) have significant overlapping mentions with the knowledge base. Therefore, Tutubalina et al. removed all the duplicates and produced refined testing splits. We follow the performance of BioSyn reported by them.

The results show that CoSiNES significantly outperforms the baselines of TF-IDF and BioBERT ranking in terms of top-k accuracy. CoSiNES achieves competitive results with BioSyn on all the datasets. Given that we didn't change any hyperparameters or architectures of CoSiNES, and directly applied the framework to new domains, we demonstrate the cross-domain applicability of CoSiNES.

	ncbi	bc5cdr-d	bc5cdr-c	bc2gm
TF-IDF@1	59.31	61.34	71.76	67.01
TF-IDF@3	69.61	69.41	76.24	76.55
TF-IDF@5	74.02	73.21	78.59	79.90
BioBERT@1	47.55	64.23	79.55	68.12
BioBERT@3	57.35	74.89	81.65	74.11
BioBERT@5	61.77	79.45	82.82	76.04
BioSyn@1	72.5	**74.1**	**83.8**	**85.8**
BioSyn@3	-	-	-	-
BioSyn@5	-	-	-	-
CoSiNES@1	**72.55**	73.52	81.65	**85.79**
CoSiNES@3	80.39	78.39	85.88	90.66
CoSiNES@5	81.37	80.52	87.76	91.68

Table 3: Results on four datasets from the biomedical domain. @1: top-1 accuracy. Here, bc5cdr-d means bc5cdr-disease and bc5cdr-c means bc5cdr-chemical.

7 Conclusion

We propose a generic, scalable, and adaptable framework CoSiNES for the entity standardization task, which maps various mentions to standard entities in the knowledge base. We first construct a new dataset ESAppMod in the technology domain and demonstrate the superiority of our framework over four other models. We conduct comprehensive experiments regarding batch size, learning rate, margin, loss calculation and different sizes of BERT, with our designed contrastive group generation and hybrid triplet mining, and show that the framework is rather robust with respect to hyper-parameters. With the optimal setting on our dataset, we further show that our model can be easily adapted to new domains with minimal changes by achieving competitive performance on four benchmark datasets from the biomedical domain covering three different types of entities.

After examining the errors produced by the framework on our proposed dataset, we categorize four different types of errors and defer to future work with the following directions: (1) integrating the framework with external knowledge. For multi-match errors, where multiple entities partially match with the mention, it would be ambiguous to retrieve the target entity. For no-match errors, external knowledge could provide extra information; (2) Adversarial training for misspellings. For technical

[4]https://huggingface.co/dmis-lab/biobert-v1.1

terms, misspelling could lead to completely different tokenization of the mentions; (3) Construct new or augment the existing training dataset with acronym samples. The pretrained language models are not specialized in recognizing acronyms. Therefore, it would be worthwhile endowing PLMs with such capability.

Limitations

We focuses on resolving various mentions from different domains. Although we have tested our framework on multiple datasets, it relies on a human-annotated dataset and effort should be taken to investigate how the model performs with emerging domains without human-annotated data. Our model works with mentions that have been extracted from raw text. It would be more practical if the model could work with raw text directly and interact with another mention-extraction module. The performance of the model is largely affected by the surface form of the mentions, although our framework is robust to variations in the surface form, it would be more beneficial to further investigate how adversarial turbulence in the mentions could affect the behaviors of the framework.

Ethics Statement

The domain and data we work with don't involve any personal information and are all publicly available. However, as the work could be potentially applied in the medical domain to resolve mentions of disease, discretion is advised when any medical decisions or diagnostics are made with the assistance of the model.

References

Mehwish Alam, Davide Buscaldi, Michael Cochez, Francesco Osborne, Diego Reforgiato Recupero, Harald Sack, Özge Sevgili, Artem Shelmanov, Mikhail Arkhipov, Alexander Panchenko, Chris Biemann, Mehwish Alam, Davide Buscaldi, Michael Cochez, Francesco Osborne, Diego Refogiato Recupero, and Harald Sack. 2022. Neural entity linking: A survey of models based on deep learning. *Semant. Web*, 13(3):527–570.

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in nli: Ways (not) to go beyond simple heuristics.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Peter Christen. 2012. *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer.

Allan Peter Davis, Cynthia J. Grondin, Robin J. Johnson, Daniela Sciaky, Roy McMorran, Jolene Wiegers, Thomas C. Wiegers, and Carolyn J. Mattingly. 2018. The comparative toxicogenomics database: Update 2019. *Nucleic Acids Research*, 47:D948 – D954.

Allan Peter Davis, Thomas C. Wiegers, Michael C. Rosenstein, and Carolyn J. Mattingly. 2012. Medic: A practical disease vocabulary used at the comparative toxicogenomics database. *Database: The Journal of Biological Databases and Curation*, 2012.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Rezarta Islamaj Dogan, Robert Leaman, and Zhiyong Lu. 2014. Ncbi disease corpus: A resource for disease name recognition and concept normalization. *Journal of Biomedical Informatics*, 47:1–10.

Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. 2018. Distributed representations of tuples for entity resolution. *Proceedings VLDB Endowment*, 11(11):1454–1467.

Shobeir Fakhraei, Joel Mathew, and José Luis Ambite. 2020. Nseen: Neural semantic embedding for entity normalization. In *Machine Learning and Knowledge Discovery in Databases*, pages 665–680, Cham. Springer International Publishing.

Shengyu Fan, Hui Yu, Xiaoya Cai, Yanfang Geng, Guangzhen Li, Weizhi Xu, Xia Wang, and Yaping Yang. 2022. Multi-attention deep neural network fusing character and word embedding for clinical and biomedical concept extraction. *Information Sciences*, 608:778–793.

Arnaud Ferré, Robert Bossy, Mouhamadou Ba, Louise Deléger, Thomas Lavergne, Pierre Zweigenbaum, and Claire Nédellec. 2020. Handling entity normalization with no annotated corpus: Weakly supervised methods based on distributional representation and ontological information. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1959–1966, Marseille, France. European Language Resources Association.

Arnaud Ferré, Louise Deléger, Robert Bossy, Pierre Zweigenbaum, and Claire Nédellec. 2020. C-norm: A neural approach to few-shot entity normalization. *BMC Bioinformatics 21 (Suppl 23)*.

Anup Kalia, Raghav Batta, Jin Xiao, Mihir Choudhury, and Maja Vukovic. 2021. Aca: Application containerization advisory framework for modernizing legacy applications. In *IEEE 14th International Conference on Cloud Computing (CLOUD)*, pages 708–710.

Adnan Khan, Sarah AlBarri, and Muhammad Arslan Manzoor. 2022. Contrastive self-supervised learning: A survey on different architectures. In *2nd International Conference on Artificial Intelligence (ICAI)*, pages 1–6.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Jiao Li, Yueping Sun, Robin J. Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J. Mattingly, Thomas C. Wiegers, and Zhiyong Lu. 2016. Biocreative V CDR task corpus: A resource for chemical disease relation extraction. *Database: The Journal of Biological Databases and Curation*, 2016.

Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. Deep entity matching with pre-trained language models. *Proceedings VLDB Endowment*, 14(1):50–60.

Alexander A. Morgan, Zhiyong Lu, Xinglong Wang, Aaron M. Cohen, Juliane Fluck, Patrick Ruch, Anna Divoli, Katrin Fundel, Robert Leaman, Jörg Hakenberg, Chengjie Sun, Heng-Hui Liu, Rafael Torres, M. Krauthammer, William W. Lau, Hongfang Liu, Chun-Nan Hsu, Martijn J. Schuemie, Kevin Bretonnel Cohen, and Lynette Hirschman. 2008. Overview of biocreative II gene normalization. *Genome Biology*, 9:S3 – S3.

Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep learning for entity matching: A design space exploration. *Proceedings of the International Conference on Management of Data*.

Hao Nie, Xianpei Han, Ben He, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. 2019. Deep sequence-to-sequence entity matching for heterogeneous entity resolution. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM, page 629–638, New York, NY, USA. Association for Computing Machinery.

George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. 2020. Blocking and filtering techniques for entity resolution: A survey. *ACM Comput. Surv.*, 53(2).

Priya Radhakrishnan, Partha Talukdar, and Vasudeva Varma. 2018. ELDEN: Improved entity linking using densified knowledge graphs. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1844–1853, New Orleans, Louisiana. Association for Computational Linguistics.

Nils Rethmeier and Isabelle Augenstein. 2022. A primer on contrastive pretraining in language processing: Methods, lessons learned and perspectives. *ACM Computing Survey*.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823.

Hamed Shahbazi, Xiaoli Z. Fern, Reza Ghaeini, Rasha Obeidat, and Prasad Tadepalli. 2019. Entity-aware elmo: Learning contextual entity representation for entity disambiguation. *ArXiv*, abs/1908.05762.

Noah A. Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 354–362, Ann Arbor, Michigan. Association for Computational Linguistics.

Mujeen Sung, Hwisang Jeon, Jinhyuk Lee, and Jaewoo Kang. 2020. Biomedical entity representations with synonym marginalization. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Elena Tutubalina, Artur Kadurin, and Zulfat Miftahutdinov. 2020a. Fair evaluation in concept normalization: a large-scale comparative analysis for BERT-based models. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6710–6716, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Elena Tutubalina, Artur Kadurin, and Zulfat Miftahutdinov. 2020b. Fair evaluation in concept normalization: A large-scale comparative analysis for BERT-based models. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6710–6716, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Ikuya Yamada, Koki Washio, Hiroyuki Shindo, and Yuji Matsumoto. 2022. Global entity disambiguation with BERT. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3264–3271, Seattle, United States. Association for Computational Linguistics.

A Biomedical Datasets Descriptions and Statistics

Detailed descriptions of the datasets can also be found in Tutubalina et al. (2020b) and Sung et al. (2020).

NCBI Disease Corpus NCBI Disease Corpus (Dogan et al., 2014) contains manually annotated disease mentions extracted from 793 PubMed abstracts and their corresponding concepts in the MEDIC dictionary (Davis et al., 2012). The July 6, 2012 version of MEDIC has 11,915 CUIs (concept ids) and 71,923 synonyms (mentions).

BioCreative V CDR BioCreative V CDR (BC5CDR) (Li et al., 2016) is a challenge for extracting chemical-disease relations. There are manual annotations for both chemical and disease from 1,500 PubMed abstracts. Like the NCBI disease corpus, disease mentions are mapped into the MEDIC dictionary. The chemical mentions are mapped into the Comparative Toxicogenomics DataBase (CTD) (Davis et al., 2018). The Nov 4, 2019 version of CTD contains 171,203 CUIs and 407,247 synonyms.

BioCreative II GN BioCreative II GN (BC2GN) (Morgan et al., 2008) contains human gene and gene product mentions from PubMed abstracts. It has 61,646 CUIs and 277,944 synonyms (Tutubalina et al., 2020a).

	KG entity	KG mention	Test mention
ncbi	12,554	73,024	204
bc5cdr-d	12,511	73,126	657
bc5cdr-c	171,284	407,600	425
bc2gm	67,370	277,944	985

Table 4: Diomedical datasets statistics. Here, KG means knowledge base, bc5cdr-d means bc5cdr-disease and bc5cdr-c means bc5cdr-chemical.

B Triplet Types

As shown in Equation 3, triplet loss is calculated based on triplets $\{x, x^+, x^-\}$, which always consist of two samples from the same class and a third sample from a different class. We usually call x the anchor of the triplet, x^+ the positive sample, and x^- the negative sample. The intuition behind the loss function is that the distance $\mathrm{d}(x, x^-)$ between the anchor and negative should be larger than the distance $\mathrm{d}(x, x^+)$ between the anchor and positive by a *margin*. The *margin* is a hyperparameter that needs to be tuned.

$$\mathcal{L} = \max(\mathrm{d}(x, x^+) - \mathrm{d}(x, x^-) + \mathrm{margin}, 0) \quad (3)$$

Based on the difference between $\mathrm{d}(x, x^-)$ and $\mathrm{d}(x, x^+)$, we can classify triplets into three categories.

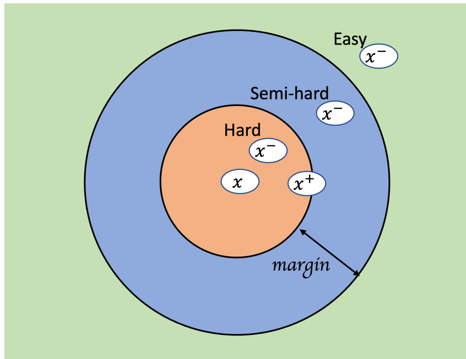

Figure 7: Different types of triplet samples.

- Easy triplets, which have a loss of zero based on Equation 2. Therefore, easy triplets provide no learning signal to the model.

$$\mathrm{d}(x, x^-) - \mathrm{d}(x, x^+) > \mathrm{margin}$$

- Semihard triplets, which have a loss less than the *margin*.

$$0 < \mathrm{d}(x, x^-) - \mathrm{d}(x, x^+) < \mathrm{margin}$$

- Hard triplets, which are most informative for the model.

$$\mathrm{d}(x, x^-) - \mathrm{d}(x, x^+) < 0$$

C Hyperparameter Search

We have done the following hyperparameter search grid on ESAppMod

Batch Size	4, 8, 16, 32
Learning Rate	1e−3, 1e−4, 1e−5, 1e−6
Margin	0.5, 1, 2, 5, 10

Table 5: Hyperparameter search on ESAppMod