Imposing Relation Structure in Language-Model Embeddings Using Contrastive Learning

Christos Theodoropoulos
KU Leuven
christos.theodoropoulos@kuleuven.be

James Henderson
Idiap Research Institute
james.henderson@idiap.ch

Andrei Catalin Coman
EPFL, Idiap Research Institute
andrei.coman@idiap.ch

Marie-Francine Moens
KU Leuven
sien.moens@kuleuven.be

Abstract

Though language model text embeddings have revolutionized NLP research, their ability to capture high-level semantic information, such as relations between entities in text, is limited. In this paper, we propose a novel contrastive learning framework that trains sentence embeddings to encode the relations in a graph structure. Given a sentence (unstructured text) and its graph, we use contrastive learning to impose relation-related structure on the token-level representations of the sentence obtained with a CharacterBERT (El Boukkouri et al., 2020) model. The resulting relation-aware sentence embeddings achieve state-of-the-art results on the relation extraction task using only a simple KNN classifier, thereby demonstrating the success of the proposed method. Additional visualization by a tSNE analysis shows the effectiveness of the learned representation space compared to baselines. Furthermore, we show that we can learn a different space for named entity recognition, again using a contrastive learning objective, and demonstrate how to successfully combine both representation spaces in an entity-relation task.

1 Introduction

Pretrained language models (LMs), such as BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019) and GPT-3 (Brown et al., 2020), capture contextualized information effectively and are used in a wide variety of natural language processing (NLP) tasks. They have revolutionized NLP research. The main mechanism of these models is multi-head self-attention (Vaswani et al., 2017), which enables capturing patterns of semantic and syntactic interest in text. However, their ability to encapsulate high level semantic information, such as relations in the text, and domain-specific knowledge, is limited because they are trained on very large corpora using the main objectives of language modeling. In many NLP tasks, pretrained LM embeddings are used as model input. A common strategy is to concatenate the embeddings that are extracted from different LMs and let the model decide which part of the information is useful for the task. This empirical approach does not provide strong intuition and results in poor explainability capabilities because most of the task-specific models are black boxes.

In this study, we present a novel contrastive learning (CL) framework to leverage the embedding space of CharacterBERT and impose a relation structure on the embeddings. The proposed framework receives a sentence and a graph that represents the text relations in a structured way, and the CL paradigm is applied to impose this structure on the token embeddings of the CharacterBERT text encoder. Different graph formulations that represent the text relations are explored. The main goal is to create a common embedding space where relations can be easily detected. To evaluate progress towards this goal, we use the ADE dataset (Gurulingappa et al., 2012), which is widely used for relation extraction (RE) (Zhao and Grishman, 2005; Jiang and Zhai, 2007; Sun et al., 2011; Plank and Moschitti, 2013) and named entity recognition (NER) tasks (Curran and Clark, 2003; Florian et al., 2006; Nadeau and Sekine, 2007; Florian et al., 2010) in the challenging field of information extraction (IE) from biomedical text.

To evaluate the efficacy of our approach, a simple baseline neural network classifier for RE, us-

Proceedings of the 25th Conference on Computational Natural Language Learning (CoNLL), pages 337–348
November 10–11, 2021. ©2021 Association for Computational Linguistics

ing the pretrained CharacterBERT medical version representations, is trained. The representations of the CharacterBERT tuned version after applying CL are used to train the same classifier, which vastly outperforms the baseline classifier. A tSNE (Van der Maaten and Hinton, 2008) analysis illustrates that meaningful relation-related clusters can be identified in the learned embedding space. This provides a second strong indication that structure can be effectively imposed on LM embeddings using our proposed framework.

Even if the main focus of this work is not solving the IE problem directly, to further explore the capabilities of the relation-aware representation space, we train a simple KNN classifier for RE that is competitive with state-of-the-art performance. Strict evaluation (Bekoulis et al., 2018b; Taillé et al., 2020) of the RE task presupposes correct detection of the boundaries and the entity type of each argument in the relation. Hence, we apply the CL paradigm to learn a distinct embedding space for the entities and use a KNN classifier to solve the NER task. Finally, we perform a strict evaluation of the complete entity-relation extraction task. This transparent, computationally inexpensive and intuitively simple approach has comparable results to the state-of-the-art models. This achievement illustrates how informative and meaningful the learned embedding spaces are.

In summary, our key contributions are:

- We propose a novel CL framework for imposing a relation-related structure on LM embeddings.
- We investigate different ways to model texts and graphs and show the effectiveness of embedding relations in pairs of token embeddings.
- We exploit the capabilities of the learned representation spaces by using them in the IE task and achieve competitive results to state-of-the-art models, even if we use transparent and intuitively simple KNN classifiers.

The paper is structured as follows. Section 2 presents the ADE dataset and the data preprocessing steps, and section 3 explains the framework in detail. In section 4, we evaluate the quality of the framework in baseline setups. The tSNE analysis is presented in section 5. In section 6, we use the framework to solve the IE task and compare the results to state-of-the-art models.

2 Dataset

This study focuses on biomedical text, and ADE dataset is used. The sentences are annotated with labels for drugs and adverse effects, as well as the relations among these entities. Adverse effects (AEs) cover a range of signs, symptoms, diseases, disorders, abnormalities, organ damage and even death caused by that drug. The corpus is annotated at the sentence-level, so non-local relations (between entities of different sentences) do not exist.

2.1 Data Preprocessing

The input of the main CL framework consists of the encoded padded sentence and the relation graph, which is extracted from the sentence. The graphs are used only in the training setup. To prepare the input for CharacterBERT, tokenization is applied to each sentence using the character-CNN module (Peters et al., 2018). The BERT tokenizer handles out-of-vocabulary (OOV) words by splitting these words into word pieces. However, the existence of word pieces can be an obstacle in creating and testing the CL experiments of this study from the implementation point of view. Additionally, word pieces may add biases to the model (El Boukkouri et al., 2020), especially in biomedical text where most of the drugs and many adverse effects are OOV words. Hence, CharacterBERT is chosen instead of BERT.

For each sentence, a knowledge graph is obtained to model the relations between the drugs and the adverse effects. The graph nodes are initialized with embeddings that are extracted by the final layer of the pretrained medical version of CharacterBERT. The graph convolutional network (GCN) (Kipf and Welling, 2016), which is a key layer of the main proposed CL framework (Fig. 1, Fig. 2), receives two inputs: an NxF matrix (N: number of nodes, F: number of features) with the embeddings (features) of each node and an adjacency matrix NxN, which models the connections (edges) of the undirected graph. Generally, the adjacency matrices are very sparse if we consider all the tokens and create the whole graph because the relations are rare and there are many singleton nodes. Alternatively, the tokens that are part of a relation can only be used, and the essential subgraph is extracted. For example, in the sentence "*Methods: we report two cases of pseudoporphyria caused by naproxen and oxaprozin.*" There are two AE relations between AE *pseudoporphyria* and the

drugs *naproxen* and *oxaprozin*. Hence, by creating the subgraph, only these AE and drug tokens are included, and the singleton nodes (rest of the sentence tokens) are removed.

The drug and the AE entities may consist of more than one word. There are two methods to model this case. On the one hand, the whole phrase can be represented as one node in the graph by averaging the embeddings of each distinct word of the phrase. On the other hand, each node refers to the last word of the entity. For example, if the initial relation is between the drug "*gabapentin*" and the adverse effect "*renal impairment*", then in the graph, the relation [gabapentin, impairment] is modeled. The latter approach is mainly adopted in nonspan-based relation extraction models (Bekoulis et al., 2018b; Zhao et al., 2020). In this study, the second approach is adopted because it gives the flexibility in applying contrastive learning at the token and relation levels.

The normalization of the adjacency matrix is essential for aggregating and propagating the information in the graph effectively (Kipf and Welling, 2016) and is described by the following equations:

$$A_{hat} = A + I, \tag{1}$$

$$A_{norm} = D^{-0.5} * A_{hat} * D^{-0.5}, \tag{2}$$

where A is the initial adjacency matrix, I is the identity matrix and D is the degree matrix.

Initially, the whole corpus is stored in one text file. Hence, the data should be transformed and stored using a different more flexible format. For each sentence of the dataset, a distinct JSON file is created and contains a list with the tokens [1], a list with named entity (NE) tags adopting the BIO encoding scheme (Sang and Veenstra, 1999; Ratinov and Roth, 2009), a list with token index pairs that are members of an existing relation, the padded encoded version of the sentence, the attention mask vector of the sentence, a list with the embeddings of each node of the graph and the normalized adjacency matrix.

2.2 Dataset Statistics

The ADE dataset is not officially split into training, validation, and test sets. Hence, we evaluate our models using 10-fold cross-validation similar to Li et al. (2017). We use the same splits as Eberts and Ulges (2020). As Taillé et al. (2020) stresses, many works on the IE task do not report the data preprocessing and detailed statistics of the datasets. This is an obstacle for a sanity check and reproducibility. The ADE dataset consists of 4,272 sentences, with 5,063 drug entities (1,048 unique drugs), 5,776 AE entities (2,983 unique AEs) and 6,821 relations. We report the statistics of each split (Table 1) and propose using this particular split for a fair comparison [2].

Split	Training Set		Test Set	
	Relation Count	Entity Count	Relation Count	Entity Count
1	6,155	9,769	666	1,070
2	6,097	9,713	724	1,126
3	6,133	9,748	688	1,091
4	6,164	9,771	657	1,068
5	6,173	9,785	648	1,054
6	6,089	9,713	732	1,126
7	6,155	9,768	666	1,071
8	6,117	9,754	704	1,085
9	6,133	9,760	688	1,079
10	6,173	9,770	648	1,069
Mean	6,139	9,755	682	1,084

Table 1: Statistics of 10-fold splits - ADE dataset

3 Framework

In essence, contrastive learning is a paradigm for learning representations which capture some auxiliary information by training them to distinguish positive from negative instances of this auxiliary information. Our framework is inspired by the recent publications on image view-based CL of visual representation (Khosla et al., 2020; Zhang et al., 2020; Henaff, 2020; Chen et al., 2020; He et al., 2020), but differs from the existing work by the application of CL to the graph and text modalities. Our work is also inspired by the semantic bootstrapping hypothesis (Pinker, 1996), which proposes that children acquire their native language through exposure to sentences of the language (i.e., a language model) paired with structured representations of their meaning (Abend et al., 2017).

The main CL framework for imposing relation-aware structure on the token embeddings is tested under two different settings. The difference in each setting is related to the modeling of the graph and the level of applying the CL paradigm. To solve the end-to-end IE task, a second model is proposed for learning a distinct embedding space where the named entities are projected.

[1] The sentence tokenization is performed using the SpaCy library.

[2] To facilitate further research, the preprocessed data and the code will be publicly available in the official repository of the paper.

3.1 Model Architectures

In the first setting (Fig. 1), we apply the CL method to the embeddings of graph nodes in their graph context and the embeddings of sentence tokens in their sentence context. We call this variation in the main CL framework *CLGS*. The positive and sampled negative graph representations are computed by a graph convolutional network (GCN) (Kipf and Welling, 2016; Schlichtkrull et al., 2018) layer followed by a pooling layer. We model the graph considering only the tokens that are part of a relation (subgraphs). To obtain one representation for the graph, average and maximum pooling strategies are tried. Tanh (range: [-1, 1]) is chosen as the activation function of the GCN layer because the text encoder also extracts negative embeddings. Hence, a similar range of embedding values should be extracted from the graph. The sentence is passed to the text encoder (CharacterBERT), which has the first six layers frozen. CharacterBERT is initialized with the pretrained weights (medical version). A pooling layer follows, to create a representation for the whole sentence. Taking the average, maximum embedding vector and the [CLS] token representation are tested as pooling strategies. The addition of a projection layers before applying CL is a common approach (Chen et al., 2020; Zhang et al., 2020). ReLU is used as the activation function of the projection layers to introduce nonlinearity. By adding the projection layers, there is the danger that the task will be solved mainly in the projection layers, while the final goal is pushing structured relation-aware information in the text encoder. Finally, CL is applied to the resulting pair of graph and sentence representations, so that the pooled sentence token embeddings are trained to carry the information in the pooled graph node embeddings.

In the second setting, we apply the CL method to the embeddings of graph relations and the embeddings of pairs of sentence tokens. This variation in the CL framework is called *CLDR*. The graph is simplified to the extreme level. Each relation is modeled completely independently in the graph, and the relation representations are extracted by concatenation of the nodes that are connected in the disjoint graphs (Fig. 2). This graph modeling makes the CL at the relation level a more tractable task. In addition, sampling negative graphs can be implemented more easily in a more controlled way. In this setting, because the graphs only have two nodes, the adjacency matrix should not be normalized in a balanced way. If the adjacency matrix is $\begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix}$, then the final node embeddings will be the same for the two nodes that form the graph. Hence, we suggest focusing more on the self-loop of each node to keep its predefined contextualized information up to a certain level [3]. The final adjacency matrix has the following format $\begin{pmatrix} \lambda & 1-\lambda \\ 1-\lambda & \lambda \end{pmatrix}$, where λ is a hyperparameter of the model. The λ parameter defines the balance of focusing on the self-loop of each node and its neighbor (connected node). Intuitively, a λ value equal to 0.8 is a good choice for focusing attention on the self-loop and having distinct embeddings for the connected nodes. ReLU is used as the activation function of the GCN layer.

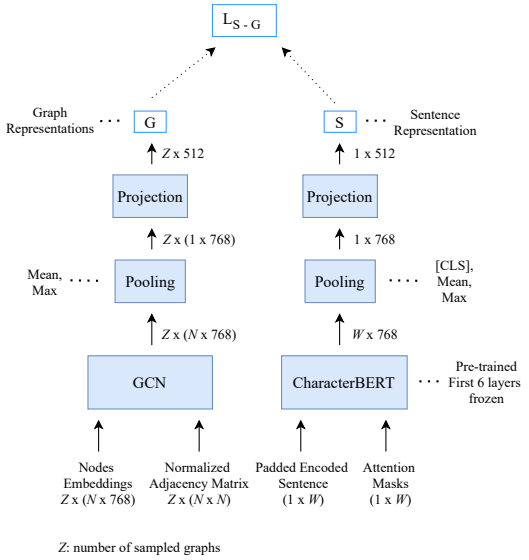

Figure 1: CL framework *CLGS* - 1st Setting

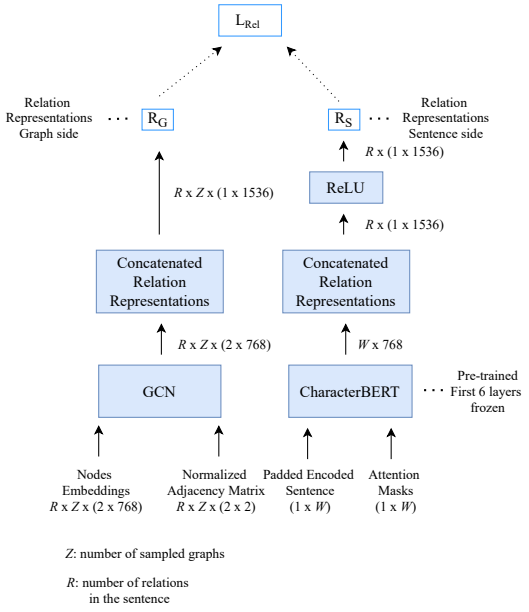

Figure 2: CL framework *CLDR* - 2nd Setting

[3] We remind that the nodes are initialized with embeddings extracted from the pretrained CharacterBERT medical version.

On the text side, the pair of tokens that form a relation in the disjoint graphs are chosen, and the concatenation of their representations is used as the final relation representation. Finally, CL is applied on the relation level, so that the pairs of sentence token embeddings are trained to carry the information in the pairs of related graph node embeddings.

A distinct model (called *CLNER*) for learning meaningful representations for named entities is designed (Fig. 3). CharacterBERT captures contextualised information very well. Hence, only one dense layer is added after CharacterBERT. Then a random sampling for the named entities is performed in a balanced way. A pool of sampled entities of the batch is selected and CL is applied on the token level.

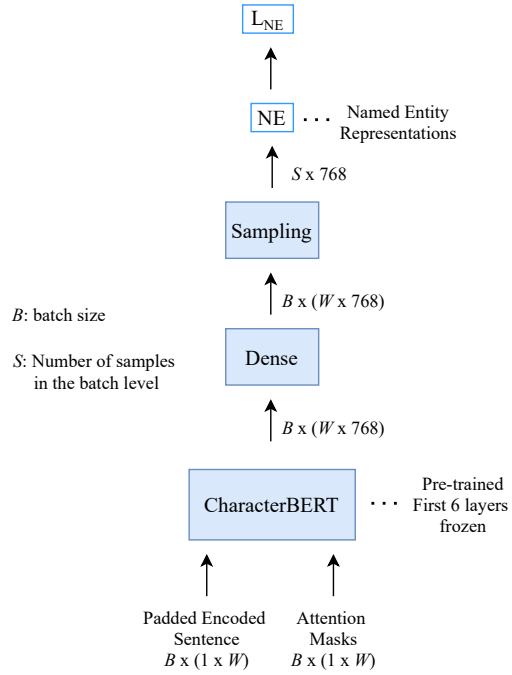

Figure 3: Model *CLNER* for learning named entity representations

3.2 Sampling Strategy

Hard negative sampling is important to effectively apply the CL paradigm. The negative graphs are created by randomly selecting tokens that are not part of an adverse effect entity, keeping the correct drug tokens, and vice versa. Hence, hard incorrect drug and adverse effects relation pairs are introduced to the graph. The positive and negative graphs of each sentence have the same number of relations but not necessarily the same number of nodes. The sampling strategy is similar for the *CLGS* (Fig. 4) and the *CLDR* model (Fig. 5). For the *CLDR* model, the positive graph is simplified to a disjoint graph, and then hard negative sampling is performed.

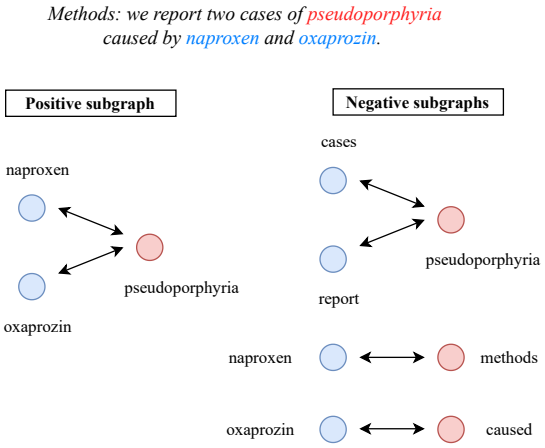

Figure 4: Example of sampling negative graphs - *CLGS* model

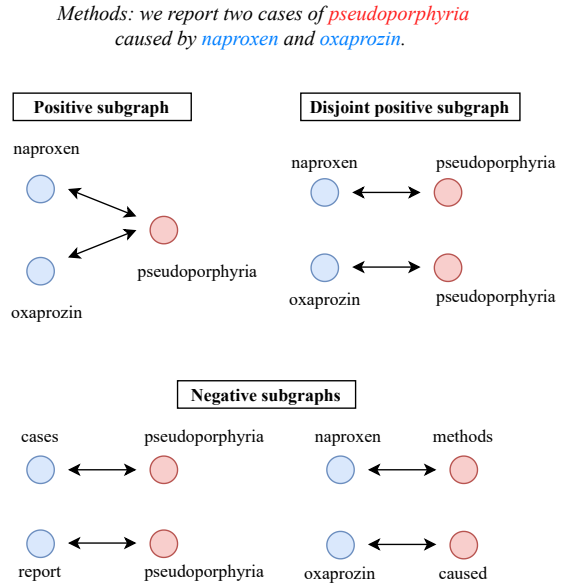

Figure 5: Example of sampling negative graphs - *CLDR* model

In the *CLNER*, random sampling[4] is executed at the batch level. Analysis of the number of different entity tags (drug, AE or outside token) in the batch is performed a priori to choose an appropriate number of positive and negative samples (balanced sampling).

[4]Hard negative sampling based on the Euclidean distance and cosine similarity is also tested, but the performance is not increased. Hence, the complexity-performance trade-off leads us to finally select random sampling.

3.3 Design Choices

In this subsection, the justification for the model design choices is discussed. For the *CLGS* and *CLDR* models, the GCN layer is the key element because it can produce useful node representations considering the graph links. The propagation rule of the GCN layer is described by the following equation:

$$X_{l+1} = \sigma(A_{norm} * X_l * W_l), \quad (3)$$

where $\sigma(\cdot)$ is the activation function (e.g., ReLU, Tanh), A_{norm} is the normalized adjacency matrix (Eq. 2), X_l the node embeddings and W_l the weights of the *l* layer.

In the first setting (*CLGS* model), the graph is propagated through the GCN layer, and a final pooled graph representation is extracted. We hypothesize that using the CL paradigm, the model can learn which part of the information is essential for the relation representations by keeping the structure-related information in the graph representation. In the second setting (*CLDR* model), the level of abstraction is reduced because instead of applying CL in the graph sentence, we use the CL paradigm at the relation level. The strategy of creating disjoint graphs results in learning similar representations for the drug and AE nodes. To address this, the relations are represented asymmetrically as a concatenation of the nodes. We hypothesize that relation-related information can be imposed in the pair-of-tokens embeddings of the LM by applying CL to them and these pair-of-nodes embeddings of the graph relations (Fig. 2).

3.4 Training Details

The models are trained using a CL loss function that is similar to the SimCLR loss function (Chen et al., 2020). In the first setting (*CLGS* model), the main concept is to leverage the two graph and sentence representations so the true representation pair is close and similar in the learned embedding space. At each training time, a set of Z graphs (the positive and some negative graphs) and the corresponding sentence are passed on the model, and the corresponding representations are calculated. Therefore, the contrastive loss receives the graph and sentence representations and for the i-th pair is as follows:

$$l_i^{(S \to G)} = -log(\frac{exp(<S_i, G_i>/\tau)}{\sum_{z=1}^{Z} exp(<S_i, G_z>/\tau)}), \quad (4)$$

where $< S_i, G_i >$ represents the cosine similarity and τ is a temperature parameter.

In the second setting (*CLDR* model), the pair of node embeddings that are extracted from the disjoint graphs encode their relation, because this is the main functionality of the GCN layer. Hence, the main idea is to increase the similarity between the representations of the correct relation in the graph and the relation representations that are extracted from the text encoder.

The contrastive loss for each sentence is as follows:

$$l^{(RS \to RG)} = \sum_{r=1}^{R} -log(\frac{exp(<RS_r, RG_r>/\tau)}{\sum_{z=1}^{Z} exp(<RS_r, RG_z>/\tau)}), \quad (5)$$

where R is the total number of relations in the sentence, RS is the relation representation of the text encoder and RG is the relation representation of the graph.

For the *CLNER* model, the contrastive loss is as follows:

$$l_{NE} = \sum_{n=1}^{N} -log(\frac{\sum_{p=1}^{P} exp(<RN_n, RN_p>/\tau)}{\sum_{k=1}^{K} exp(<RN_n, RN_k>/\tau)}), \quad (6)$$

where N is the total number of tokens in the batch, P is the number of the positive samples (same NE tag), K is the total number of samples and RN is the extracted token representation.

We use a batch-size of 8 for training the *CLGS* and *CLDR* models, and 16 for the *CLNER* model. ADAM optimizer (Kingma and Ba, 2014) is selected with a learning rate of 1e-5 [5].

4 Evaluation - Baseline

For the *CLGS* model, the first evaluation step is a simple similarity check. We use the trained *CLGS* model to extract the sentence representation and the positive and negative graph representations for all the sentences in the test set. Then, a similarity check is applied using the extracted sentence and graph representations. The most similar graph is predicted as the positive sentence graph. Given the positive and all the negative hard graphs extracted from each sentence, the model should be able to detect the correct graph. The different model variations perform well, but the mean pooling selection in the graph and sentence side results in better performance, as the accuracy is over 91%. The addition of the projection layers is not advantageous.

[5] More information about hyperparameter tuning-selection is given in the Appendix section.

Graph Pooling	Text Pooling	Projection layer	Accuracy
Mean	[CLS]	-	88.39
Mean	Mean	-	**91.23**
Max	Max	-	89.1
Mean	[CLS]	Yes	88.63
Mean	Mean	Yes	87.68

Table 2: Results - *CLGS* model: Finding the correct graph with similarity check

The second evaluation step is applied to both models (*CLGS* and *CLDR*). Following previous research on representation learning (Henaff, 2020; Chen et al., 2020; He et al., 2020; Zhang et al., 2020), we evaluate the tuned CharacterBERT text encoder, taken from the trained *CLGS* and *CLDR* models, in a linear classification setting, where all the candidate relations (concatenation of the token embeddings) are created, and a linear classification layer is trained for the RE task. As a baseline model, we use the pretrained medical CharacterBERT to create the representation for the relations[6]. This linear setting directly provides insight into how successfully the relation-related structure is imposed at the token level of the text encoder, by evaluating the quality of the learned representations for RE.

Model	Precision	Recall	F1
Baseline	69.96	64.39	66.79
$\text{CharacterBERT}_{\text{CLGS}}$	56.82	59.42	58.09
$\text{CharacterBERT}_{\text{CLDR}}$	**79.51**	**84.39**	**81.73**

Table 3: RE - linear classification setting

Using the tuned CharacterBERT representation from the *CLGS* model (mean graph and text pooling) results in poor performance. The pooling layer smooths the information. Hence, structure-related information cannot be passed at the token level of the text encoder. A smarter pooling strategy that preserves most of the relation-aware information would be ideal, but designing such pooling is difficult. The main obstacle is the varied number of relations. In contrast, when we use the tuned CharacterBERT of the *CLDR* model, the basic classifier vastly outperforms the baseline model. This is a strong indication that the relation-related structure is successfully imposed on the pairs of token embeddings of the text encoder.

[6]We also try fine-tuning both the text encoder and the linear head, but the performance is not improved.

5 tSNE Analysis

A tSNE analysis is performed to further explore the quality of the learned embedding spaces. Using the tuned CharacterBERT of the *CLDR* model, the relation representation space is created. We project the positive (orange dots) and hard negative relations (blue dots), where one of the two relation tokens is correct. In the tSNE plot (Fig. 6), meaningful relation clusters can be easily identified, which demonstrates the efficiency of our framework (*CLDR* model). The relation representations are asymmetric, as the drug and AE tokens have similar representations (Fig. 7). This means that we cannot solve RE and NER tasks using the same representation space. Hence, we learn a different space for the named entities (*CLNER* model).

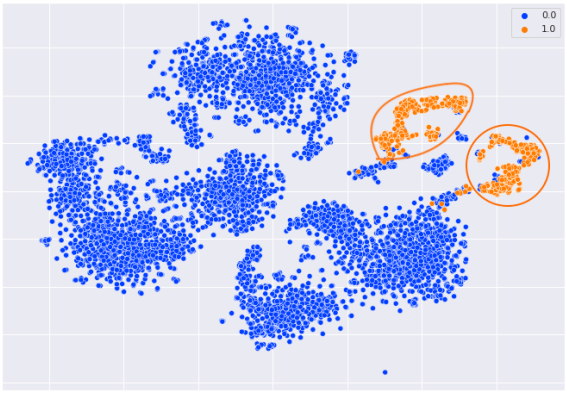

Figure 6: tSNE plot - Relation representation space obtained with CharacterBERT of *CLDR* model (1: relation, 0: no relation)

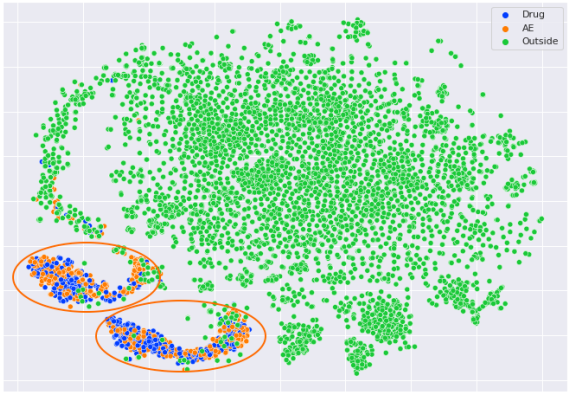

Figure 7: tSNE plot - Relation representation space obtained with CharacterBERT of *CLDR* model - Named Entities

In the tSNE plot in the entity representation space (Fig. 8), we can detect insightful entity clusters. In particular, the clusters related to the drug tags (B-DRUG, I-DRUG) are very dense and well

shaped. This is a strong finding that illustrates that the *CLNER* model can extract very good representations for the NER task.

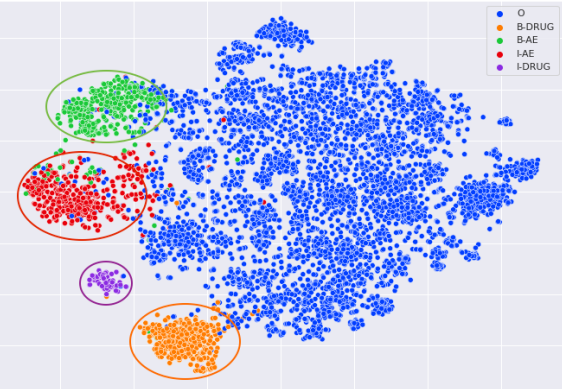

Figure 8: tSNE plot - Entity representation space obtained with *CLNER* model

6 Entity-Relation task

The insights of the tSNE analysis, with the well-defined clusters in the embedding spaces, lead us to approach the entity-relation task using intuitively simple and transparent KNN classifiers. For the RE task, we utilize the tuned CharacterBERT of the *CLDR* model to create the candidate relation representations. At the inference step, for each candidate relation, we decide whether it is positive based on the labels of the k-nearest neighbors in the learned embedding space. The value of k is chosen based on the performance in the randomly selected validation set (10% of training set) for each fold. We adopt the same strategy for the NER task using the *CLNER* model and project each token to the named entity representation space.

To solve both NER and RE tasks, we combine the two semantic spaces. First, we determine whether a candidate relation (concatenation of the tokens) is predicted as positive in the relation representation space, which is obtained by the tuned CharacterBERT of the *CLDR* model. Then, we determine whether the boundaries and the types of the two entities in the candidate relation are predicted correctly in the entity representation space obtained by the *CLNER* model. All possible candidate relations and the named entities of the test set are classified.

We strictly evaluate the performance of the IE task. As Bekoulis et al. (2018b) state, an entity is considered correct if its boundaries are detected correctly and the predicted type (drug or AE) matches the ground truth. In the same setup, a relation is considered correct if its type and the two entities (boundaries and type) involved in the relation are correctly predicted. We measure precision, recall and F1 score. Following previous work on IE, we report the macro-averaged F1 score, and as 10-fold cross-validation is adopted, we average the scores over the folds.

Model	NER	RE	RE-
Li et al., 2016	79.5	63.4	-
Li et al., 2017	84.6	71.4	-
Bekoulis et al., 2018b	86.4	74.58	-
Bekoulis et al., 2018a	86.73	75.52	-
Tran and Kavuluru, 2019	87.11	77.29	-
Eberts and Ulges, 2020	89.25	79.24	-
Wang and Lu, 2020	89.7	80.1	-
Zhao et al., 2020	89.4	81.14	-
Ours	88.3	79.97	86.5

Table 4: Test set results: macro-averaged F1 score

Table 4 presents the best performing models, evaluated on the ADE (Gurulingappa et al., 2012) dataset. These studies address the IE problem as a joint task, solving NER and RE tasks jointly. Li et al. (2016) employ global features and a CNN (LeCun et al., 1995) module to solve the problem. The proposed model of Li et al. (2017) includes bidirectional RNNs (Graves et al., 2013), inspired by the work of Miwa and Bansal (2016). Bekoulis et al. (2018a,b) formulate the IE problem as a multi-head selection problem. Tran and Kavuluru (2019) approach the IE task as a table-filling problem and introduce a relation-metric network, combining the idea of metric learning and the usage of CNNs for table filling. Eberts and Ulges (2020) present a span-based model that its core module is pre-trained BERT (Devlin et al., 2018). Wang and Lu (2020) propose table-sequence encoders that learn table and sequence representations to solve the IE problem. Zhao et al. (2020) introduce a deep cross-modal attention network, constructed by stacking multiple attention units, for joint entity and relation extraction.

In the RE task, we achieve very competitive results using a simple and transparent KNN classifier. In contrast, the state-of-the-art models (Wang and Lu, 2020; Zhao et al., 2020) are very complex and computationally expensive. This fact highlights the high quality of the learned relation representation space (*CLDR* model). In principle, the NER task is a sequence-tagging problem. However, we obtain good performance with a KNN classifier

that performs the inference in the learned entity representation space (*CLNER* model).

Notably, the last column of Table 4 (*RE-*) presents the performance of the RE KNN classifier in predicting whether there is a relation between two tokens, without considering the NER task (type and boundaries of the entities). In this case, the F1 score is 86.5, and this value is the upper bound performance of our approach. Hence, incorporating a state-of-the-art model for the NER task (e.g., Wang and Lu, 2020, Eberts and Ulges, 2020) could further improve the scores of the RE task under strict evaluation. However, we use the SpERT model (Eberts and Ulges, 2020) for NER (F1 score: 89.25), but the results in the RE task are not improved. This illustrates that our NER results are already very competitive.

The above results reveal the quality of the representations for both NER and RE tasks. Hence, the proposed CL framework can be used as a preprocessing and representation learning step in the pipeline for IE models. The CL framework can be trained to leverage the embedding space and create meaningful, disentangled representations for the IE task. We successfully evaluated the representations with a simple KNN classifier, but the learned representations can be used as input in complex models for entity and relation classification to achieve better results and faster convergence. We will explore this research direction in the future.

7 Conclusion

We present a novel CL framework, which, in principle, is text encoder-agnostic, for effectively imposing relation-related structure to LMs and leveraging the embedding space. We evaluate the quality of the learned representations using relative baselines and competitively solve an entity-relation task. The overall results indicate that the learned representations are very powerful. The performed tSNE analysis illustrates that meaningful clusters can be easily identified in the learned embedding spaces. We note that the proposed framework can be used as a representation learning step for complex IE systems. In future work, we intend to explore the capabilities of our approach in continual learning settings and exploit external graph structured knowledge in representation learning of language data.

Acknowledgments

This work is supported by the Research Foundation – Flanders (FWO) and Swiss National Science Foundation (SNSF). Christos Theodoropoulos and Marie-Francine Moens are affiliated to Leuven.AI - KU Leuven institute for AI, B-3000, Leuven, Belgium.

References

Omri Abend, Tom Kwiatkowski, Nathaniel J Smith, Sharon Goldwater, and Mark Steedman. 2017. Bootstrapping language acquisition. *Cognition*, 164:116–143.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018a. Adversarial training for multi-context joint entity and relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2830–2836.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018b. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems with Applications*, 114:34–45.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

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

James R Curran and Stephen Clark. 2003. Language independent ner using a maximum entropy tagger. In *Proceedings of the Seventh Conference on Natural language Learning at HLT-NAACL 2003*, pages 164–167.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Markus Eberts and Adrian Ulges. 2020. Span-based joint entity and relation extraction with transformer pre-training. pages 2006–2013.

Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Jun'ichi Tsujii. 2020. Characterbert: Reconciling elmo and bert for word-level open-vocabulary representations from characters. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6903–6915.

Radu Florian, Hongyan Jing, Nanda Kambhatla, and Imed Zitouni. 2006. Factorizing complex models: a case study in mention detection. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 473–480.

Radu Florian, John F Pitrelli, Salim Roukos, and Imed Zitouni. 2010. Improving mention detection robustness to noisy input. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 335–345.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee.

Harsha Gurulingappa, Abdul Mateen Rajput, Angus Roberts, Juliane Fluck, Martin Hofmann-Apitius, and Luca Toldo. 2012. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *Journal of Biomedical Informatics*, 45(5):885–892.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Olivier Henaff. 2020. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pages 4182–4192. PMLR.

Jing Jiang and ChengXiang Zhai. 2007. A systematic exploration of the feature space for relation extraction. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 113–120.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Yann LeCun, Yoshua Bengio, et al. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995.

Fei Li, Meishan Zhang, Guohong Fu, and Donghong Ji. 2017. A neural joint model for entity and relation extraction from biomedical text. *BMC bioinformatics*, 18(1):1–11.

Fei Li, Yue Zhang, Meishan Zhang, and Donghong Ji. 2016. Joint models for extracting adverse drug events from biomedical text. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 2838–2844. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Steven Pinker. 1996. *Language Learnability and Language Development: With New Commentary by the Author*, volume 7. Harvard University Press.

Barbara Plank and Alessandro Moschitti. 2013. Embedding semantic similarity in tree kernels for domain adaptation of relation extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1498–1507.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155.

Erik F Tjong Kim Sang and Jorn Veenstra. 1999. Representing text chunks. In *Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 173–179.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.

Ang Sun, Ralph Grishman, and Satoshi Sekine. 2011. Semi-supervised relation extraction with large-scale word clustering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 521–529.

Bruno Taillé, Vincent Guigue, Geoffrey Scoutheeten, and Patrick Gallinari. 2020. Let's stop error propagation in the end-to-end relation extraction literature! In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 3689–3701.

Tung Tran and Ramakanth Kavuluru. 2019. Neural metric learning for fast end-to-end relation extraction. *arXiv preprint arXiv:1905.07458*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Jue Wang and Wei Lu. 2020. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 1706–1721. Association for Computational Linguistics.

Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. 2020. Contrastive learning of medical visual representations from paired images and text. *arXiv preprint arXiv:2010.00747*.

Shan Zhao, Minghao Hu, Zhiping Cai, and Fang Liu. 2020. Modeling dense cross-modal interactions for joint entity-relation extraction. pages 4032–4038.

Shubin Zhao and Ralph Grishman. 2005. Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 419–426.

Training Details - Hyperparameters

Initially, the data are split to train and test set using 10-fold cross-validation (same splits to Eberts and Ulges (2020)). In each training session, for each split, the seed number is set to 42 in order to randomly create a validation set for tuning (10% of the train set). The seed number is chosen in order to have the same split of train and validation set in all training sessions of the CL framework and the baseline and KNN classifiers. ADAM optimizer (Kingma and Ba, 2014) is selected with learning rate 1e-5 for training of the CL framework. The best weights, based on the performance in the validation set, are stored.

We train the CL framework (*CLGS*, *CLDR*, *CLNER* models) for 20 epochs and apply the technique of early stopping after 3 epochs without improvement in the validation set. We experiment with different hyperparameter values and select the best values based on the performance in the validation set (averaged across the 10 splits) in the basic classifier that is presented in section 4. For the *CLGS* and *CLDR* models, the different negative graphs of each sentence are created offline. The length of sentences varies significantly, so the number of negative graphs also varies. Based on that, randomly selecting 8 negative graphs for each training set is intuitively a good choice. In parentheses, there are the tested values. The hyperparameters of the *CLGS* model are:

- Batch Size: 8 (8, 16)
- Temperature τ: 0.1 (0.05, 0.1, 0.2)
- Number of negative graphs: 8 (4, 8, 12)

Those of the *CLDR* model are:

- Batch Size: 8 (8, 16)
- λ parameter (adjacency matrix): 0.8 (0.7, 0.8, 0.9)
- Temperature τ: 0.1 (0.05, 0.1, 0.2)
- Number of negative graphs: 8 (4, 8, 12)

The essential parameter of the *CLNER* model is the number of samples. The number of available tokens depends on the batch size. In order to sample in a balanced way (Table 5), when the batch size is 16, a good number of samples is 80. For example, if we have a *'B-DRUG'* token, then we sample all the tokens with the same tag (positive tokens - around 20, Table 5) and the remaining negative tokens are sampled in a balanced way. This sampling strategy should be defined because the NE token distribution is highly imbalanced (Table 6). The *'O'* tag is highly represented, while the *'I-DRUG'* tag is under-represented. The temperature value τ is set to 0.1.

The KNN classifier has only one hyperparameter, the number of k neighbors that are taken into account in the inference step. We choose the k value based on the performance in the validation set for each split. The k value is 5 for the RE KNN classifier, and 7 for the NER KNN classifier (section 6).

NE type	Count
B-DRUG	19
I-DRUG	4
B-AE	21
I-AE	26
O	268

Table 5: Average number of tokens per NE tag - Batch size: 16

NE type	Count
B-DRUG	5,039
I-DRUG	1,062
B-AE	5,701
I-AE	7,054
O	71,858

Table 6: Total number of tokens per NE tag