Enriching Contextualized Representations with Biomedical Ontologies: Extending KnowBert to UMLS

Guilhem Piat, Nasredine Semmar, Alexandre Allauzen, Hassane Essafi, Julien Tourille

Enriching Contextualized Representations with Biomedical Ontologies: Extending KnowBert to UMLS

Guilhem Piat[1], Nasredine Semmar[1], Alexandre Allauzen[2], Hassane Essafi[1], and Julien Tourille[1]

[1] Université Paris-Saclay, CEA, List
F-91120, Palaiseau, France
{guilhem.piat, nasredine.semmar, hassane.essafi, julien.tourille}@cea.fr
[2] Université Paris Dauphine, LAMSADE
F-75775, Paris Cedex 16, France
alexandre.allauzen@dauphine.psl.eu

Abstract. Currently, biomedical document processing is mostly human work. Software solutions which attempt to alleviate this burden exist but generally do not perform well enough to be helpful in many applications. Concurrently, there exist projects which organize concepts in the biomedical field. Therefore, we seek to leverage existing structured knowledge resources to improve biomedical language modeling. In this paper, we provide an implementation integrating the UMLS knowledgebase into a BERT-based language model, aiming to improve its performance in biomedical Named Entity Recognition. To achieve this, we extend KnowBert, a recently developed technique for integrating knowledge into language models. Preliminary results reveal the challenges of applying KnowBert to the biomedical domain given the number and subtlety of different concepts in UMLS. Going forward, addressing these challenges and combining this with other approaches such as BioBERT may help expand the range of usefully automatable biomedical language processing tasks.

Keywords: Artificial neural networks, Knowledge based systems, Knowledge representation, Machine learning, Biomedical informatics, Information Extraction

1 Introduction

With over a million articles published every year in the biomedical field and the large number of patient records generated by hospitals, it is increasingly difficult for healthcare professionals to keep up to date on research, carry out systematic reviews, or search for patient information. There is thus a demand for language processing tools able to identify and extract meaningful information from these texts.

For this reason, multiple knowledge bases such as the Unified Medical Language System (UMLS) and the OpenTargets LIterature coNcept Knowledge base

(LINK) have been created to make information more searchable. Our objective is to enable the recently developed Transformer-based pretrained neural Language Models (LMs) [1] to make explicit use of this knowledge, in order to improve their performance and interpretability.

We follow the method described by Peters et al. [2] known as KnowBert to integrate knowledge derived from the UMLS Knowledge Base into a BERT-based language model. We thus call this model KnowBert-UMLS.

Other projects such as BioBERT [3] and ClinicalBert [4] have successfully specialized language models to the biomedical domain. However, their approach has typically not explicitly leveraged structured knowledge sources such as UMLS. A notable exception is UmlsBERT [5], which leverages UMLS as a thesaurus to explicitly teach BERT synonymy and enriches biomedical word representations with rough clustering.

Our approach using KnowBert differs significantly in that it makes use of the full vocabulary of UMLS to enrich word representations, and jointly performs entity linking. It is also fairly indifferent to the pretrained LM used as a base, and can be combined with another specialized model such as BioBERT.

In the context of large and specialized knowledge bases such as UMLS, we find the approach proposed by Peters et al. [2] to be computationally unrealistic with current tools for most organizations. Preliminary biomedical Named Entity Recognition evaluations of our model trained on a small subset of our training corpus demonstrate a decrease in performance with respect to models with non-enriched word representations. We investigate the reasons for this, and propose ways to alleviate this computational burden.

The remainder of the paper is organized as follows. We contextualize our approach and motivations by discussing related work in Section 2. In Section 3, we overview the architecture of KnowBert and discuss the specifics of our extension of it to the UMLS Knowledge Base. The preliminary experimental results are reported and discussed in Section 4. Finally, we present in Section 5 our conclusions and future work.

2 Related Work

In the wake of the advent of the Transformer architecture introduced by Vaswani et al. [1], language processing tasks have increasingly been handled by neural language models based upon this architecture such as BERT [6]. Due to the differences between the language used in the biomedical field and the types of text typically used to train these models, many projects have sought to leverage the Transformer architecture in the more specific and rigorous biomedical context. The typical approach has been to pre-train language models on specialized text as has been done with BioBERT [3], BioMed-RoBERTa [7], SciBert [8], and Clinical BERT [4], which all incorporate various amounts and proportions of biomedical text in the pre-training phase of their models.

However, not only does this type of pre-training usually lead models to underperform on general-domain text as demonstrated by Arumae and Bhatia

[9], large models with attention mechanisms such as BERT are also notoriously computationally expensive to pre-train. Furthermore, the ability for a model to associate concepts (*e.g.* "*COVID-19*" and "*respiratory failure*") is predicated on these concepts appearing in the pre-training corpus, leading to difficulty adapting to some forms of distributional shift. These limitations have lead to interest in different methods of adapting these models to specific domains.

One such method is integrating information derived from existing Knowledge Bases (KBs) into these models. There have been multiple methods proposed to achieve this. E-BERT [10], for instance, projects entity embeddings derived from the KB to the input word embedding space. UmlsBERT [5], on the other hand, explicitly adds semantic group embeddings to words found in UMLS. ERNIE [11] and BERT-MK [12] learn a fused representation of contextualized word and entity representations. K-Adapters [13] are a cheaper alternative to the ERNIE process which can't suffer from catastrophic forgetting by using Adapters [14]. One drawback of these methods, particularly in the biomedical context, is that they all require a separate upstream Entity Linking (EL) step to be made at inference.

While Entity Linkers are known not to be perfectly reliable, Biomedical Entity Linking is a particularly difficult task. Some of the best performing models include RysannMD [15], which achieves 0.436 F1 on the CRAFT corpus, and the dual encoder architecture proposed by Bhowmik, Stratos, and Melo [16] which achieves 0.564 F1 on the MedMentions corpus [17]. The effectiveness of the knowledge integration step being inherently limited by the Entity Linking step, the possibility of performing knowledge enrichment without or jointly with Entity Linking becomes attractive.

KEPLER [18] is one such model which introduces a knowledge embedding loss as an objective for language model pre-training, aligning contextual word representations with entity description representations. As such, it does not require an EL step at inference. The KnowBert model developed by Peters et al. [2], on the other hand, grafts a KB-specific entity linking module into a transformer-based pretrained LM such as BERT, in order to jointly perform Entity Linking and contextualized word representation enrichment, making it also a standalone method requiring no upstream EL with the additional benefit of explicitly identifying the entities present in the text.

Our approach to biomedical Language Modeling thus differentiates itself from existing methods by leveraging the ability of KnowBert to jointly perform Entity Linking and Language Modeling, and applying it in a biomedical context in order to improve word representations and generate metadata which can be used for a variety of downstream tasks. Additionally, relying on structured knowledge enables KnowBert-based models to recognize new concepts without additional training and to perform similarly to non-specialized models on general domain text.

3 KnowBert-UMLS

As shown in Fig. 1, the KnowBert architecture is composed of three main components: a pretrained Language Model backbone, a Knowledge Attention and Recontextualization Module (or KAR) which performs Entity Linking and knowledge enrichment of word representations, and a candidate mention generator.

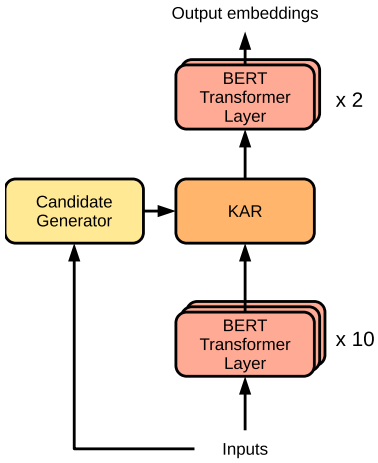

Fig. 1. Abstraction of the KnowBert architecture. KnowBert extends BERT by adding a Knowledge Attention and Recontextualization module (KAR) between two transformer layers; in this case between layers 10 and 11.

3.1 Pretrained BERT

While the KnowBert method can apply to most Transformer-based pretrained language models, we focus on BERT as it was used by Peters et al. [2]. BERT models comprise L Transformer layers. For a sequence of N tokens, each layer i takes as input an $N \times H$-dimensional sequence representation $\mathbf{H}_{i-1}$ and outputs a representation $\mathbf{H}_i$ which integrates more contextual information by applying a multi-headed attention mechanism to $\mathbf{H}_{i-1}$ followed by a Multi Layer Perceptron. In the case of $\text{BERT}_{\text{BASE}}$, we have $L = 12$ and $H = 768$. The final output of each token is thus a contextualized representation in $\mathbb{R}^H$.

3.2 Ontology and Candidate Generator

The KnowBert method ties a pretrained language model to an Ontology, specifically a Knowledge Base. For our purposes, we define a Knowledge Base $\mathcal{K}$ as a set of $J_{\mathcal{K}}$ entities e_j, each with a vectorial representation $\mathbf{e}_j \in \mathbb{R}^K$. We use the

UMLS Knowledge Base, with each entity corresponding to a Concept Unique Identifier. The entity embeddings we use are computed according to the adversarial method provided by Maldonado, Yetisgen, and Harabagiu [19] with $K = 50$.

To perform the Entity Linking step, the KAR requires a candidate generator to create a list $\mathcal{C}$ of candidate mentions. Specifically, each sequence is associated to a set $\mathcal{S}$ of S *candidate spans*, which may or may not contain an entity mention. Each candidate span is then assigned a corresponding list of *candidate entities*, including a null entity representing the lack of an entity mention within the candidate span. Formally, we have:

$$\mathcal{C} = \{(s, \{e_{s,1}, \ldots, e_{s,J_s}\}) | s \in \mathcal{S}\} \tag{1}$$

with each candidate span s being associated to a set of J_s candidate entities, and each entity e_j having a corresponding vector $\mathbf{e}_j \in \mathbb{R}^K$.

These candidates are produced by a candidate generator which follows rules specific to the KB being used. KnowBert as specified by Peters et al. [2] implements compatibility with two KBs, namely WordNet and Wikipedia.

The challenge in crafting a mention generator for the biomedical domain, and specifically UMLS, is the variability of formulations for each concept. In the case of UMLS, each concept is associated to a list of common strings (called "atoms") that may represent it. For instance, the concept for lung cancer is associated to 97 different forms, including "pulmonary carcinoma".

To leverage these atoms, we have attempted several methods based on string similarity and cosine similarity of vectorial word representations. We have found the most effective option for our purpose to be the QuickUMLS python library [20] which, given some text, identifies candidates in the form `(span_start, span_end, concept_ID)`. We then aggregate candidate entities by candidate span, derive an empirical estimate of the prior probabilities for each entity from MedMentions, and find the relevant entity embeddings as described in Fig. 2. Finally, we feed the output of this candidate generation process to the KAR.

In practice, matching each of the approximately 180M (million) sequences in our training corpus to the 16M atoms in UMLS on-demand is prohibitively computationally expensive. In order to achieve this, we precompute the candidates for each of our sequences ahead of time and create a lookup table for each file in our corpus. This needs to be done only once and is parallelizable, but nonetheless 3.5% of our corpus took six days to process across seven nodes of a computing cluster, each equipped with two Xeon 36-thread processors with a clock speed of 3GHz, and required us to favor speed over recall and precision when considering QuickUMLS settings.

Depending on which similarity measure and threshold are chosen, QuickUMLS trades off between recall and execution time. We settled on Jaccard similarity, with a threshold of 0.7 as the best compromise we could find.

In our experience, the computational impact at inference is fairly low for on-demand low-volume applications, as the candidate generator typically takes fractions of a second to process a sequence.

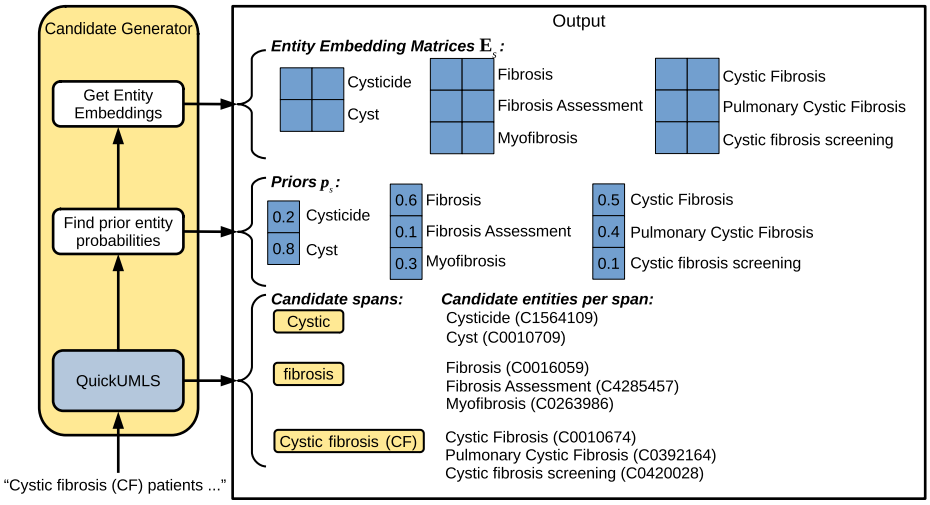

Fig. 2. Detailed structure and output of the UMLS candidate generator.

3.3 KAR

The KnowBert approach adds a KB-specific "Knowledge Attention and Recontextualization module", or KAR, between two transformer layers in a pretrained BERT model. This module is a relatively inexpensive addition to the pretrained model, with in our case only approximately 0.3% as many trainable parameters as $\text{BERT}_{\text{BASE}}$.

Multiple KBs can be used in tandem: theoretically, a KAR can be inserted between every pair of layers in the transformer. In practice, the insertion of a KAR too close to the input layer causes too much perturbation to the flow of information and prevents the model from recovering during training. As suggested by Peters et al. [2], in order to minimize the language model's perplexity[1], we insert the KAR between the tenth and eleventh layers of $\text{BERT}_{\text{BASE}}$ as per Fig. 1.

This module performs entity linking on the intermediate contextualized word representations and pools them with the relevant entity embeddings. This results in contextualized word representations which are enriched with information extracted from a KB. Specifically, the KAR takes as input a sequence representation $\mathbf{H}_i$ and a list $\mathcal{C}$ of S candidate mentions as generated by the Candidate Generator (see (1)).

As described by Peters et al. [2] and illustrated in Fig. 3, the KAR first linearly projects the output of the previous transformer layer $\mathbf{H}_i$ to the entity embedding space:

$$\mathbf{H}_i^{proj} = \mathbf{H}_i \mathbf{W}^{proj} + \mathbf{b}^{proj} \tag{2}$$

[1] Perplexity is computed as the exponential of the cross-entropy loss, and is a standard measure of how well the language model predicts samples.

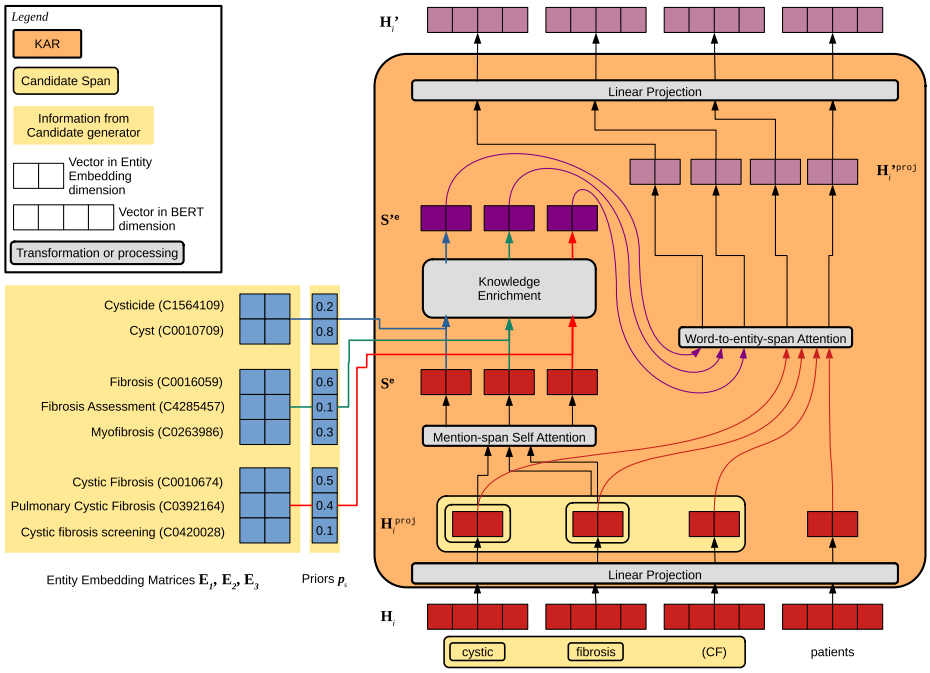

Fig. 3. Detailed structure of the Knowledge Attention and Recontextualization module (KAR).

Where $\mathbf{W}^{proj}$ and $\mathbf{b}^{proj}$ are learned.

Then, the projected embeddings for the words in each span are pooled into a matrix $\mathbf{S} \in \mathbb{R}^{S \times K}$ of span embeddings. Each span embedding is computed following "End-to-end neural coreference resolution" by Lee et al. [21], who describe a way to compute text span vectors: each token in each span is associated to a weight computed from the contextualized embeddings fed through a trained FFNN. These weights are softmaxed with respect to each span of text, and serve as the weights for a weighted-sum pooling of the non-contextualized token embeddings, resulting in non-contextualized text span embeddings.

These span embeddings are then contextualized with a standard transformer layer to allow the entity linker to identify relationships between entity mentions, resulting in the contextualized span embedding matrix $\mathbf{S^e}$.

$$\mathbf{S^e} = \text{MLP}(\text{MultiHeadAttn}(\mathbf{S}, \mathbf{S}, \mathbf{S})) \tag{3}$$

Where MLP and MultiHeadAttn designate a position-wise Multi-Layer Perceptron and a Multi-Headed Attention layer respectively.

The contextualized span embedding $\mathbf{s^e}$ of every candidate span s is then used to pool the corresponding matrix of candidate entity embeddings $\mathbf{E}_s$ from the KB, resulting in a predicted entity representation:

$$\vec{\psi_s} = \text{Softmax}(\text{MLP}(\mathbf{p}_s, \mathbf{s^e} \cdot \mathbf{E}_s))$$

$$\tilde{\mathbf{e}}_s = \vec{\psi_s} \cdot \mathbf{E}_s \tag{4}$$

where $\mathbf{p}_s \in \mathbb{R}^{J_s}$ is the vector of prior probabilities for the candidate entities associated with span s, and $\vec{\psi_s} \in \mathbb{R}^{J_s}$ is an estimate of their posterior probabilities.

The predicted entity representation embeddings $\tilde{\mathbf{e}}_s$ of each span s are packed and added to contextualized span embeddings $\mathbf{S^e}$, forming the knowledge-enriched span embedding matrix $\mathbf{S'^e}$:

$$\mathbf{S'^e} = \mathbf{S^e} + \tilde{\mathbf{E}} \tag{5}$$

$\mathbf{H'}_i^{proj}$ is computed with word-to-enriched-entity-span attention, similarly to applying a regular transformer layer to $\mathbf{S'^e}$ but substituting the query in the attention mechanism for projected word embeddings $\mathbf{H}_i^{proj}$:

$$\mathbf{H'}_i^{proj} = \text{MLP}(\text{MultiHeadAttn}(\mathbf{H}_i^{proj}, \mathbf{S'^e}, \mathbf{S'^e})) \tag{6}$$

Finally, the knowledge enriched contextual word representation output of the KAR is a projection of $\mathbf{H'}_i^{proj}$ back to BERT contextualized word representation space with an added skip connection:

$$\mathbf{H'}_i = \mathbf{H'}_i^{proj}\mathbf{W'}^{proj} + \mathbf{b'}^{proj} + \mathbf{H}_i^{proj} \tag{7}$$

where $\mathbf{W'}^{proj}$ and $\mathbf{b'}^{proj}$ are learned.

The linked entity for span s is simply $e_{s,\text{argmax}(\vec{\psi_s})}$.

3.4 Training

There are three training steps for KnowBert models. First, once the mention generator is written, the KAR is trained on the Entity Linking task on spans given by the corpus, minimizing a log-likelihood loss for the predicted probability distribution over candidate entities:

$$\mathcal{L}_{\text{EL}} = -\sum_s log\left(\frac{\exp(\psi_{sg})}{\sum_{k=1}^{n}\exp(\psi_{sk})}\right) \tag{8}$$

with ψ_{sg} the score for the ground truth entity in $\vec{\psi_s}$.

The second training phase involves continuing the pre-training of BERT using both a Masked Language Model and a Next Sentence Prediction objective. This phase corrects the disruptions incurred by the Language Model when grafting the KAR between the Transformer Layers in BERT. This step also adjusts the weights of the KAR for Entity Linking, minimizing:

$$\mathcal{L}_{\text{KnowBert}} = \mathcal{L}_{\text{BERT}} + \mathcal{L}_{\text{EL}} \tag{9}$$

We call this phase the "re-training" step to differentiate it from the BERT pre-training step and the fine-tuning step.

The final step, as with most pretrained LMs, is to fine-tune it to the target task.

4 Preliminary Experiments

We present the preliminary results of our experiments, which intend to highlight the challenges that must be overcome to successfully apply the KnowBert method to the biomedical domain with the UMLS Knowledge Base. We use the same pretrained backbone for our KnowBert-UMLS model as Peters et al. [2] in their original paper on KnowBert, *i.e.* English $\text{BERT}_{\text{BASE}}$ uncased.

4.1 Masked LM and Next Sentence Prediction

For a large source of raw biomedical text, we scraped the PubMed Central database of Open Access articles and processed them for next sentence prediction using the tool provided with the source code for "Knowledge Enhanced Contextual Word Representations" by Peters et al. [2]. At the end of this training phase, KnowBert can be used as a typical pretrained BERT model.

Due to time constraints, we were unable to generate candidates for the approximately 180M sequences in the corpus, and had to limit our re-training corpus to approximately 6M sequences. As shown in Table 1, this lack of re-training data has prevented the language model from successfully integrating the KAR, with a masked LM perplexity several orders of magnitude larger than $\text{BERT}_{\text{BASE}}$, $\text{BERT}_{\text{LARGE}}$, and the KnowBert models produced by Peters et al. [2].

Table 1. Masked Language Model perplexity for both BERT models, the KnowBert variants produced by Peters et al. [2], and KnowBert-UMLS.

Model	Perplexity
$\text{BERT}_{\text{BASE}}$	5.5
$\text{BERT}_{\text{LARGE}}$	4.5
KnowBert-Wiki	4.3
KnowBert-Wordnet	4.1
KnowBert-W+W	3.5
KnowBert-UMLS	10387.7

4.2 NER

We choose to fine-tune KnowBert-UMLS on the Biomedical Named Entity Recognition task on the n2c2 corpus, previously known as i2b2 2010 [22], with an 80% - 20% split between training and validation sets using cross-entropy loss. In Table 2, we compare our performance versus four BERT-based models, namely BioBERT [3], clinicalBERT [4], BlueBERT [23] and $\text{BERT}_{\text{BASE}}$, all fully fine-tuned on the NER task with the same linear classifier architecture. The performance of our various baselines were taken from Fraser et al. [24].

Examples of correct and incorrect predictions made by KnowBert-UMLS, formatted according to the IOB2 standard, can be found in tables 3 and 4 respectively. The example in table 4 is a quite typical incorrect prediction, as it consists of a span that overlaps with the correct span and has a correct label. This type of error is the most common, constituting 32% of the model's mistakes. Many of these mistakes are ambiguous even to humans – for instance, the matter of having to include the token "Estimated" in the "blood loss" entity is not self-evident. We perform a complete breakdown of error types as specified by Fraser et al. [24] in table 5.

Table 2. Performance of BERT-based language models on the n2c2 NER task, measured as Micro-averaged strict Precision, Recall and F1. Results for BioBERT, clinicalBERT and BlueBert from Fraser et al. [24].

Model	P	R	F1
$\text{BERT}_{\text{BASE}}$	0.85	0.87	0.86
BioBERT	0.86	0.88	0.87
clinicalBERT	0.87	0.88	0.88
BlueBERT	0.88	0.90	0.89
KnowBert-UMLS	0.80	0.81	0.80

Table 3. Example of correct NER predictions by KnowBert-UMLS pulled from the n2c2 evaluation set.

Sequence	status	post	total	abdominal	hysterectomy	and	bilateral	salpingo-oophorectomy	.
True	O	O	B-treatment	I-treatment	I-treatment	O	B-treatment	I-treatment	O
Predicted	O	O	B-treatment	I-treatment	I-treatment	O	B-treatment	I-treatment	O

Table 4. Example of incorrect NER prediction by KnowBert-UMLS pulled from the n2c2 evaluation set.

Sequence	Estimated	blood	loss	was	100	cc
True	B-problem	I-problem	I-problem	O	O	O
Predicted	O	B-problem	I-problem	O	O	O

While the contextualized word representations contain enough information for the classification model to perform significantly better than chance, our results reveal a decrease in performance with respect to a non-modified $\text{BERT}_{\text{BASE}}$. This is further demonstration of the fact that the re-training procedure is the

Table 5. Breakdown of types of mistakes made by KnowBert-UMLS in proportion of total prediction mistakes made.

Error type	Proportion (%)
Correct label	
Overlapping span	32.0
Incorrect label	
Overlapping span	10.2
Correct span	15.1
False positive	29.1
False negative	13.6

performance bottleneck and requires more text than our candidate generator can realistically process in a reasonable time frame.

Our evaluation is performed with SeqEval [25] in strict mode. Like the results from Fraser et al. [24], its metrics are on an entity-level rather than at token-level, meaning that a true positive is a fully matching mention span. A predicted mention that overlaps with a true mention but is not identical counts as a false positive and a false negative.

5 Conclusions

Successfully integrating UMLS knowledge into a pretrained LM using the KnowBert method presents a significant challenge due to the size of the knowledge base and the difficulty of generating candidate mentions. Our candidate generator based on QuickUMLS was not able to generate candidates with enough efficiency and precision to make re-training possible at the required scale. We are currently working on generating candidates for larger chunks of the re-training corpus in order to evaluate the progress made by Knowbert-UMLS as a function of corpus size, and make projections on its performance when trained on the full dataset.

In order to successfully re-train KnowBert-UMLS, the candidate generator must be improved significantly. Its main source of false negatives is the introduction of abbreviations of long terms in the beginning of the text which are subsequently re-used. These abbreviations are often absent from the UMLS and cannot be identified by the generator. Solving this issue would likely increase recall significantly when identifying candidate spans. This may allow a different recall/time compromise to be found within QuickUMLS settings.

Regardless of possible improvements to recall however, deploying this at scale, whether for re-training or practical text processing purposes, is likely to remain prohibitively slow for most individuals and organizations. Future work will involve finding a more effective and computationally efficient approach to tackle candidate generation, for instance as a machine learning problem or with a fast NER-based span pre-selection step.

Furthermore, whilst we chose to evaluate the performance of KnowBert-UMLS using $\text{BERT}_{\text{BASE}}$ as a backbone to isolate the effect of the KAR, the KnowBert method has the advantage of being compatible with other approaches such as BioBERT, clinicalBERT, BlueBERT, or SciBert. In addition to the potential performance improvements on biomedical tasks, these pretrained models may be less expensive to re-train due to the potentially smaller distributional shift between pre-training, KAR training, and re-training corpora.

In addition to the improvements that need to be made to make KnowBert-UMLS competitive, there are a number of potential ways to enhance it and expand its range of applicability.

Multiple Knowledge Bases As shown by Peters et al. [2], KnowBert is capable of accommodating multiple KARs for multiple KBs simultaneously. Depending on the practical application, it could be useful to develop a KnowBert model combining UMLS with WordNet, Wikipedia, YAGO [26], or other specialized KBs. It would also be interesting to assess the performance of one such model in order to understand to what extent multi-specialization is possible.

Re-training with Adapters Adapters, as proposed by Houlsby et al. [14], have seen some success for efficiently fine-tuning pretrained LMs such as BERT. It is conceivable that this approach may aid in the re-training process by reducing the number of parameters to train, and may help reduce the memory footprint of KnowBert in some practical applications. Specifically, in cases that involve multiple knowledge bases or sets of knowledge bases used independently from each other, such an approach may allow one copy of a pretrained LM to be loaded into memory whilst the relevant set of KARs and adapters can be applied as a function of the token sequence being processed.

References

[1] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[2] Matthew E. Peters et al. "Knowledge Enhanced Contextual Word Representations". In: *EMNLP*. 2019.

[3] Jinhyuk Lee et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* (Sept. 2019), btz682. ISSN: 1367-4803, 1460-2059. DOI: 10.1093/bioinformatics/btz682.

[4] Emily Alsentzer et al. "Publicly Available Clinical BERT Embeddings". In: *Proceedings of the 2nd Clinical Natural Language Processing Workshop*. 2019, pp. 72–78.

[5] George Michalopoulos et al. "UmlsBERT: Clinical Domain Knowledge Augmentation of Contextual Embeddings Using the Unified Medical Language System Metathesaurus". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2021, pp. 1744–1753.

[6] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019, pp. 4171–4186.

[7] Suchin Gururangan et al. "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020, pp. 8342–8360.

[8] Iz Beltagy, Kyle Lo, and Arman Cohan. "SciBERT: A Pretrained Language Model for Scientific Text". 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)*. 2019, pp. 3615–3620.

[9] Kristjan Arumae and Parminder Bhatia. "CALM: Continuous Adaptive Learning for Language Modeling". In: *arXiv preprint arXiv:2004.03794* (2020).

[10] Nina Poerner, Ulli Waltinger, and Hinrich Schütze. "E-BERT: Efficient-Yet-Effective Entity Embeddings for BERT". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*. 2020, pp. 803–818.

[11] Zhengyan Zhang et al. "ERNIE: Enhanced Language Representation with Informative Entities". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 1441–1451.

[12] Bin He et al. "Integrating Graph Contextualized Knowledge into Pretrained Language Models". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*. 2020, pp. 2281–2290.

[13] Ruize Wang et al. "K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters". In: *arXiv:2002.01808 [cs]* (Dec. 2020).

[14] Neil Houlsby et al. "Parameter-efficient transfer learning for NLP". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 2790–2799.

[15] John Cuzzola, Jelena Jovanović, and Ebrahim Bagheri. "RysannMD: A biomedical semantic annotator balancing speed and accuracy". en. In: *Journal of Biomedical Informatics* 71 (July 2017), pp. 91–109. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2017.05.016.

[16] Rajarshi Bhowmik, Karl Stratos, and Gerard de Melo. "Fast and Effective Biomedical Entity Linking Using a Dual Encoder". In: *Proceedings of the 12th International Workshop on Health Text Mining and Information Analysis*. 2021, pp. 28–37.

[17] Sunil Mohan and Donghui Li. "MedMentions: A Large Biomedical Corpus Annotated with UMLS Concepts". In: *arXiv:1902.09476 [cs]* (Feb. 2019).

[18] Xiaozhi Wang et al. "KEPLER: A unified model for knowledge embedding and pre-trained language representation". In: *Transactions of the Association for Computational Linguistics* 9 (2021), pp. 176–194.

[19] Ramon Maldonado, Meliha Yetisgen, and Sanda M Harabagiu. "Adversarial learning of knowledge embeddings for the unified medical language system". In: *AMIA Summits on Translational Science Proceedings* 2019 (2019), p. 543.

[20] Luca Soldaini and Nazli Goharian. "Quickumls: a fast, unsupervised approach for medical concept extraction". en. In: *MedIR workshop, sigir*. 2016, pp. 1–4.

[21] Kenton Lee et al. "End-to-end Neural Coreference Resolution". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2017, pp. 188–197.

[22] Özlem Uzuner et al. "2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text". In: *Journal of the American Medical Informatics Association* 18.5 (2011), pp. 552–556.

[23] Yifan Peng, Shankai Yan, and Zhiyong Lu. "Transfer Learning in Biomedical Natural Language Processing: An Evaluation of BERT and ELMo on Ten Benchmarking Datasets". In: *Proceedings of the 18th BioNLP Workshop and Shared Task*. 2019, pp. 58–65.

[24] Kathleen C Fraser et al. "Extracting UMLS Concepts from Medical Text Using General and Domain-Specific Deep Learning Models". In: *EMNLP-IJCNLP 2019* (2019), p. 157.

[25] Hiroki Nakayama. *seqeval: A Python framework for sequence labeling evaluation*. 2018. URL: https://github.com/chakki-works/seqeval.

[26] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. "Yago: A Core of Semantic Knowledge". In: *16th International Conference on the World Wide Web*. 2007, pp. 697–706.