TTI-COIN at BioCreative VII Track 2

Fully neural NER, linking, and indexing models

Tomoki Tsujimura, Ryuki Ida, Isanori Oiwa, Makoto Miwa, and Yutaka Sasaki
Toyota Technological Institute, Nagoya, Aichi, Japan

***Abstract*—We built neural models that extract chemical entities from full papers, link them to database entries, and select key terms from the extracted terms. All our models adopt pretrained BERT language models, i.e., SciBERT and PubMedBERT, enabling the training of high-performance models from a small amount of data.**

***Keywords*——linking; indexing; neural network; representation learning; TF-IDF**

I. Introduction

Chemical names are one of the most searched entities in PubMed. Identifying and indexing chemical terms in the literature can greatly help researchers find the relevant articles, and they can also be useful for downstream NLP tasks. However, in practice, the mentions of chemical terms have various aliases and chemical formulae. Indexing them is not easy because it requires understanding the article's main topic. BioCreative VII track 2 (1, 2) is a shared task for named entity recognition, linking, and indexing from full paper articles. In the task, annotations for 150 full paper articles are provided as training examples.

In natural language processing, the use of pretrained models has recently led to high performance, and they are often employed in named entity recognition and linking. Following this line, we aim to build a high-performance neural system for chemical identification and entity linking. After the official submission period, we built a neural chemical indexing model that utilizes a bag of sentences as input, often used in distant supervision tasks (3), and compared the results with those of the TF-IDF (term frequency-inversed document frequency) model. As a result, we obtained the F-scores of 0.8284 for NER, 0.7506 for linking, and 0.3806 for indexing on the *strict setting*.

II. BioCreative VII track 2

In the BioCreative VII track 2, there are two subtasks that require information extraction from full papers: Chemical Identification and Chemical Indexing prediction.

A. Chemical Identification

Chemical Identification Subtask is composed of named entity recognition and linking. The target entities are chemical entities, and they have to be linked to the corresponding MeSH Identifiers (IDs) if they exist. Some entities have multiple corresponding MeSH IDs. As a preliminary experiment, we examined the numbers of the cases where the beginnings or endings of the entity spans did not match subword delimiters by some publicly-available pre-trained BERT tokenizers as in Tab. 1. Of course, if tokens do not fit the delimiters, the model we used in this study cannot identify the correct spans. However, from the results, such cases are very few, and they are 0.5% of the entire entities even for the PubMedBERT tokenizer that caused failures most.

B. Chemical Indexing prediction

Indexing is the task of assigning index MeSH identifiers (IDs) to articles for indexing and searching. The task focuses on selecting representative linked entities to be indexed, and this task does not require other tasks such as creating the reverse index. Key chemicals such as target chemicals and major concepts to categorize articles, which may correspond to more abstract concepts concerning the MeSH tree structure than the concepts directly appearing on the articles, are chosen as the index IDs for the articles.

To understand how the index IDs are chosen, we examined the index IDs and their correspondence to the mentions' IDs in the training corpus. Tab. 2 summarizes the statistics of the coverage for different settings of candidate IDs. Here, #Cand ID denotes the number of the candidate MeSH IDs, #Index ID denotes the covered index IDs by the candidates, and Positive denotes the ratio of #Index ID to #Cand ID. "Parents" and "grandparents" represent the relationships on the MeSH tree structure. For the concepts in the supplementary concept records, we define the parent relationship with their heading mappings. The MeSH tree structure is a forest consisting of several subtrees, and one MeSH ID may appear in multiple trees, so one MeSH ID may have multiple parents.

Since the concepts appearing in the indices should not be highly abstract, we check if we can reduce the candidates by removing the highly abstract concepts. Specifically, we exclude concepts that are in the shallow part of the MeSH tree. This reduction, however, affects both the number of candidates and covered indices, and this turns out not to be always effective.

III. Method

A. Named Entity Recognition

We build a simple BERT-based NER model that outputs BILOU labels. First, the hidden vectors of the BERT final layer are fed to the output layer with the dropout ratio of 0.2. Then, the softmax function is applied to the output to obtain a probability distribution for the BILOU labels. We employ the SciBERT as the pre-trained BERT model since the number of missing spans is sufficiently small as in Tab. 1 and its subword spans match our linking model. The training is performed with Adam with a learning rate of 5e-5 for 5 epochs. We use a batch size of 32. The input raw text is processed by the SciBERT's tokenizer; if the input text length exceeds the maximum length of 510 ([BOS], [EOS] excluded), the text is split with the stride size of 128 tokens. Finally, we decode the output by the Viterbi algorithm.

B. *Linking*

We employ the neural linking model that we built in the n2c2 2019 Track 3. The model's hyper-parameters were tuned using SciBERT (5) on the n2c2 2019 Track 3, so we decided to employ SciBERT as the pre-trained BERT model. The entity linking model consists of pre-trained SciBERT with pooling, fully connected, and cosine similarity-based output layers. Fig. 1 shows the overview of our linking model.

The input of the model is the tokenized entity mention without any context. Training instances are built from the annotated corpus and the records from a database (i.e., MeSH thesaurus). For the corpus, we extract instances according to the annotated spans of the entities. If an entity span cannot be successfully tokenized from full text, we omit the span from training instances. For the database, we extract all the synonyms from the MeSH Descriptor file (desc2021.xml) and MeSH Supplemental Concept Records file (supp2021.xml)[1]. The total numbers of instances were 30,759 from the 120 papers in the training set, 7,000 from the 30 papers in the development set, and 957,168 from MeSH thesaurus. Note that the total number of mentions differs from the total number of annotated mentions because we omitted the mentions that have failed in tokenization. Additionally, we use the whole BC5CDR data sets as the additional training instances. The total number of instances from BC5CDR is 28,785.

The model first encodes a given mention by the pretrained BERT model and obtains the pooled representation by taking the average pooling over the BERT's last hidden subword vectors except for ones for special tokens (i.e., [BOS] and [EOS]). After that, the mention vector of the input is obtained by applying a fully connected layer with a tanh activation to the pooled representation. Then the model calculates the cosine similarity between the mention vector and the learnable embedding vector of each MeSH ID at the output layer. For the out-of-vocabulary concepts (denotes as "-" in the corpus), we prepare not the embedding vector but a learnable scalar value and use it as the similarity. For the sake of simplicity, we treat combined concepts as one distinct concept. Finally, the output probabilities are obtained with the softmax function over all the similarities with temperature parameters.

The number of the total embedding vectors is about 348,000. Even with the many training instances, the training data is sparse, and hence the model tends to underfit during the training because of the large number of concept embedding vectors. The average accuracy stacks around 62% at the end of the training,

TABLE I. THE NUMBER OF SPANS FAILED TO TOKENIZE. THERE ARE 38,339 ANNOTATED MENTION SPANS IN TOTAL FOR CHEMICAL IDENTIFICATION SUBTASK.

Model	# Failure
BERT-base-uncased (4)	43
SciBERT (5)	34
BioBERT-v1.1 (6)	15
PubMedBERT (base, uncased) (7)	197

TABLE II. INDEXING STATISTICS ON THE TRAINING CORPUS

	#Cand ID	#Index ID	Positive
Annotated index	-	364	-
Annotated mentions' IDs	5,039	287	5.70%
+ Parent IDs	10,521	342	3.25%
+ limit to depth ≤ 2	10,280	341	3.32%
+ limit to depth ≤ 3	9,003	330	3.67%
+ Parents & Grandparents	14,568	344	2.36%
+ All ancestor IDs	17,845	345	1.93%
+ limit to depth ≤ 2	16,553	343	2.07%
+ limit to depth ≤ 3	12,986	331	2.55%
+ limit to depth ≤ 4	8,943	287	3.21%
+ limit to depth ≤ 5	5,471	216	3.95%

TABLE III. TUNED HYPER-PARAMETERS FOR THE NEURAL INDEXING MODEL.

Parameter	Symbol	Value	Range
Batch size		8	{8,16,32,64}
Head size	h^{l}	8	{1,2,4,8}
Learning rate		5.74652e-06	[1e-06, 1e-04]
Dropout rate	p^{ld}	0.430275	[0.0, 0.5]
Context size	l^{c}	16	{4,8,16}
Max mention size	l^{m}	13	{0,5,9,13}

TABLE IV. NER RESULTS. THE "INDIVIDUAL" ROW SHOWS THE MEAN (STDDEV) VALUES OF 5 DIFFERENT RUNS. WE SUBMITTED THE SAME NER OUTPUT FOR ALL 5 SUBMISSIONS.

Set	Run	Submission ID	Strict			Approximate		
			P	*R*	*F*	*P*	*R*	*F*
Dev	Individual		0.8443 (0.0121)	0.8139 (0.0156)	0.8286 (0.0054)	0.8988 (0.0123)	0.8677 (0.0180)	0.8827 (0.0036)
	Ensemble		0.8671	0.8305	0.8484	0.9166	0.8804	0.8982
Test	Ensemble	1-5	0.8476	0.8101	0.8284	0.9128	0.8670	0.8893

[1] https://www.nlm.nih.gov/databases/download/mesh.html

and the training accuracy for IDs with only corresponding instances from databases is about 28%. To alleviate this underfitting, we employ a trick that overwrites the embedding vectors. First, we train the model until the decrease of training losses slows down. Then, we substitute the average of the vectors of the instances from databases corresponding to each concept for the concept's embedding vector. After overwriting, we resume the training with initial Adam parameters for a few epochs. This trick increases the accuracies for both the training and development sets with limited annotations.

We employed ArcFace (8) for the loss function and Adam for the optimizer. We used hyper-parameters originally tuned on the n2c2 2019 Track3 dataset by Optuna (9). We initialized SciBERT parameters by the pretrained model "scibert-scivocab-uncased" and fine-tuned them during the training. Since the instances from the corpus were much fewer than those from the database, we performed up-sampling to make ratios of instances from each domain roughly the same. For prediction, we chose the biggest probability class as the predicted label.

C. Indexing

We used a TF-IDF model and a neural network-based model for Subtask 2 (indexing). Both models predict whether a candidate MeSH ID is an index ID or not. Based on the investigation described in subsection 2.B for each article, we used MeSH IDs that directly appear in the article as well as all the parent IDs for candidates.

1) TF-IDF model for indexing

TF-IDF was calculated using the number of occurrences of the MeSH ID for each article, and the index IDs were determined with a certain threshold to the TF-IDF value. First, the spans of a chemical that appeared in the papers were replaced by its MeSH ID. Next, we calculated IDF based according to the training corpus. Then we set a threshold value and determined the index IDs by checking whether each TF-IDF value exceeds the threshold or not. The threshold value was set differently for the MeSH IDs and the parent MeSH IDs. When a MeSH ID appeared directly or as a parent MeSH ID, it was judged to be an index ID if both of them exceeded the threshold value. The threshold value was tuned to maximize the F-score by Optuna.

2) Neural network model for indexing

Fig. 2 shows an overview of the indexing model using neural networks. Initially, for each MeSH ID directly occurring in the paper and its parent MeSH ID, a bag of sentences is created by combining the related mentions and their surrounding contexts. The sentences in this bag are a combination of the context of the previous and following l^c tokens and the mention with special tokens. For each sentence, we insert special tokens representing the beginning and ending of the mention ([BOM] and [EOM] in the figure). In this case, if the mention length exceeds a certain threshold l^m, we delete the middle token and replace it with the special token [SHORT] so that it fits into l^m (e.g., in the figure, the intention of the second sentence was "1,7-bis(4-hydroxy-3methoxyphenyl)-1,6-heptadien-3,5-dione", but it was shortened to "1,7-[SHORT] 5-dione"). If the sentence corresponds to a child of the MeSH ID of the bag, a special token [CSEP] and the entry term of the MeSH ID of the bag are given (e.g., the sentence in the bottom row of the figure originally had only "ll32", but it now has "ll32 [CSEP] curcumin"). If the entry term exceeds l^m, it is shortened using [SHORT] as in the main body of the message.

The bag of sentences created in the above process is used as input to the model. Indexing is performed by binary classification of whether the MeSH ID of the given bag should be an index ID or not. The sentence representation is obtained with a word-wise attention module over the subword vectors from the BERT encoder. The following attention module over the sentence representations is employed to obtain the bag representation, and the obtained representation is mapped to one dimension by a fully-connected layer. The indexing probability is calculated by using a sigmoid function.

TABLE V. LINKING RESULTS ON THE DEVELOPMENT SET. OW STANDS FOR OVERWRITING THE CONCEPT'S EMBEDDING VECTORS. AB3P STANDS FOR RESOLVING ABBREVIATIONS BY AB3P. BC5CDR STANDS FOR USING ADDITIONAL TRAINING INSTANCES FROM BC5CDR DATASETS. ABBREVIATION STANDS FOR RESOLVING ABBREVIATIONS BY OUR ABBREVIATION RESOLVER.

Model	**Strict**		
	P	***R***	***F***
Baseline model	0.7325	0.8812	0.8000
- OW	0.7433	0.8380	0.7878
+ Ab3P	0.7600	0.8855	0.8180
+ Ab3P, - OW	0.7704	0.8445	0.8058
+ BC5CDR	0.7400	0.8823	0.8049
+ BC5CDR, - OW	0.7635	0.8542	0.8063
+BC5CDR, Ab3P	0.7640	0.8844	0.8198
+BC5CDR, Ab3P, - OW	0.7830	0.8575	0.8186
+ BC5CDR, Abbreviation	0.7652	0.8834	0.8201

TABLE VI. LINKING RESULTS ON THE TEST SET.

Model	**Submission ID**	**Strict**			**Approximate**		
		P	***R***	***F***	***P***	***R***	***F***
Baseline model	2	0.7078	0.8698	0.7805	0.6612	0.9018	0.7554
+ Ab3P	3	0.7338	0.8683	0.7954	0.6954	0.8976	0.7760
+ BC5CDR	4	0.7038	0.8670	0.7769	0.6424	0.8961	0.7399
+ BC5CDR, Ab3P	5	0.7306	0.8658	0.7925	0.6782	0.8919	0.7625
+ BC5CDR, Abbreviation	1	0.7038	0.8670	0.7769	0.6421	0.8959	0.7395

During training, dropout is performed with probability p^{ld} for the input to the output layer. Both word- and sentence-level attention modules employ the multi-head attention (10), and for simplicity, the same number of heads h^l is used.

IV. Experimental Setting

Learning is performed by minimizing the negative log-likelihood. Adam was chosen as the learning algorithm. BERT was initialized with SciBERT. Hyperparameter tuning was done using Optuna. The list of tuned hyperparameters, their ranges, and their final values are shown in Tab. 3.

We randomly selected 30 papers out of 150 provided training examples as development examples throughout the entire task. The neural network models were implemented in Pytorch. The experimental environment was TITAN V (12GB memory), V100-DGXS (32GB memory), and GeForce 3090 RTX (24GB memory) GPUs.

V. Results and discussion

A. Named Entity Recognition

Table 4 shows NER scores for our BERT-based model. We had the ensemble output for the NER model by averaging the output probabilities of 5 different runs and then applying Viterbi decoding. We submitted the same NER output for all 5 official submissions.

B. Linking

Table 5 shows the linking scores for the *strict setting* on the development set. If we omitted the weight overwriting at the output layer, the baseline model degrades the linking performance. However, when we use additional BC5CDR corpus for training, the weight overwriting has almost no effect on the performance. The model achieved comparable F-scores even without the weight overwriting to the baseline model, probably because the model has achieved sufficient learning using the BC5CDR training examples. Thus, there is no longer a need to force learning progression by overwriting weights. Since our model makes one prediction for one mention span while there should be some mentions that have different surface forms but indicate the same concept, the recall scores of our model are relatively larger than the precision scores.

In both models, the abbreviated form resolution by Ab3P is valid, and our abbreviation solver has almost the same performance as the Ab3P. By applying the Ab3P abbreviation solver, predictions for abbreviated mentions are removed, leading to precision improvement.

Table 6 shows the test scores of the linking model. Again, the test results have the same tendency as the development set; the models trained with BC5CDR corpus show roughly the same scores for the ones without it. Using abbreviation solver improves the overall scores, while our solver (submission id=1) did not work on the test set as we expected because of the bug (on the development set, we made characters lowercase, but in the test set, we mistakenly skipped the preprocessing.)

C. Indexing

Tab. 7 and 8 show the performances of our indexing model. While the neural network-based model and the TF-IDF model worked on the development set with the gold mention spans and the concept labels, it turned much worse than the TF-IDF model on the test set. One of the reasons should be the error propagation from the NER and linking models. Since our neural indexing model was trained with gold spans and concepts, the BERT encoder might be sensitive to the wrong spans and output the broken sentence representations. On the other hand, the TF-IDF model counts the number of occurrences and ignores any contexts; thus, it is more robust for error propagation than the neural model. One possible solution is using the instances with spans and concepts labeled by the pipeline systems, but we could not have done this due to the time limitation.

VI. Conclusions

We built the pipeline of neural NER, linking, and indexing models for BioCreative VII track 2 and the TF-IDF indexing model. Our models showed the large performance gaps between the predictions from gold spans/concepts and pipeline outputs, especially for the neural indexing model. Future work is to alleviate the error propagation problem.

TABLE VII. Indexing results on the development set for the strict setting.

Model	Setting	P	R	F
Neural Network	Gold span & link	0.6557	0.5714	0.6107
	Pipeline	0.2292	0.3143	0.2651
TF-IDF	Gold span & link	0.5714	0.5714	0.5714
	Pipeline	0.3651	0.3286	0.3459

TABLE VIII. Indexing results on the test set. These are the unofficial results since we submitted these after the official deadline.

Model	Submission ID	Strict			Approximate		
		P	*R*	*F*	*P*	*R*	*F*
TF-IDF	1	0.2753	0.6163	0.3806	0.3889	0.7924	0.4865
Neural Network	3	0.2352	0.2661	0.2497	0.3418	0.4677	0.3520
Ensemble	2	0.3267	0.3276	0.3271	0.4083	0.5377	0.4189

References

1. Robert Leaman, Rezarta Islamaj and Zhiyong Lu. (2021) Overview of the NLM-Chem BioCreative VII track: Full-text Chemical Identification and Indexing in PubMed articles. Proceedings of the seventh BioCreative challenge evaluation workshop.

2. Rezarta Islamaj, Robert Leaman, David Cissel, Meng Cheng, Cathleen Coss, Joseph Denicola, Carol Fisher, Rob Guzman, Preeti Kochar, Nicholas Miliaras, Zoe Punske, Keiko Sekiya, Dorothy Trinh, Deborah Whitman, Susan Schmidt and Zhiyong Lu. (2021) The chemical corpus of the NLM-Chem BioCreative VII track: Full-text Chemical Identification and Indexing in PubMed articles. Proceedings of the seventh BioCreative challenge evaluation workshop.

3. Yankai Lin et al. (2016) Neural Relation Extraction with Selective Attention over Instances. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 2124–2133.doi:10.18653/v1/P16-1200. url:https://aclanthology.org/P16-1200.

4. Jacob Devlin et al. (2019) 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). Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. doi: 10.18653/v1/N19-1423. url: https://aclanthology.org/N19-1423.

5. Iz Beltagy, Kyle Lo, and Arman Cohan. (2019) 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). Hong Kong, China: Association for Computational Linguistics, Nov. 2019,pp. 3615–3620. doi: 10 . 18653 / v1 / D19 - 1371. url:https://aclanthology.org/D19-1371.

6. Jinhyuk Lee et al. (2020) BioBERT: a pre-trained biomedical language representation model for biomedical text mining. In: Bioinformatics 36.4 (2020), pp. 1234–1240.

7. Yu Gu et al. (2020) Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing.2020. eprint: arXiv:2007.15779.

8. Jiankang Deng et al. (2019) ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019, pp. 4690–4699.

9. Takuya Akiba et al. (2019) Optuna: A Next-generation Hy-perparameter Optimization Framework. In:The 25thACM SIGKDD Conference on Knowledge Discoveryand Data Mining (KDD' 19). 2019, pp. 2623–2631.

10. Ashish Vaswan et al. (2017) Attention is All You Need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp. 6000–6010.isbn: 9781510860964.

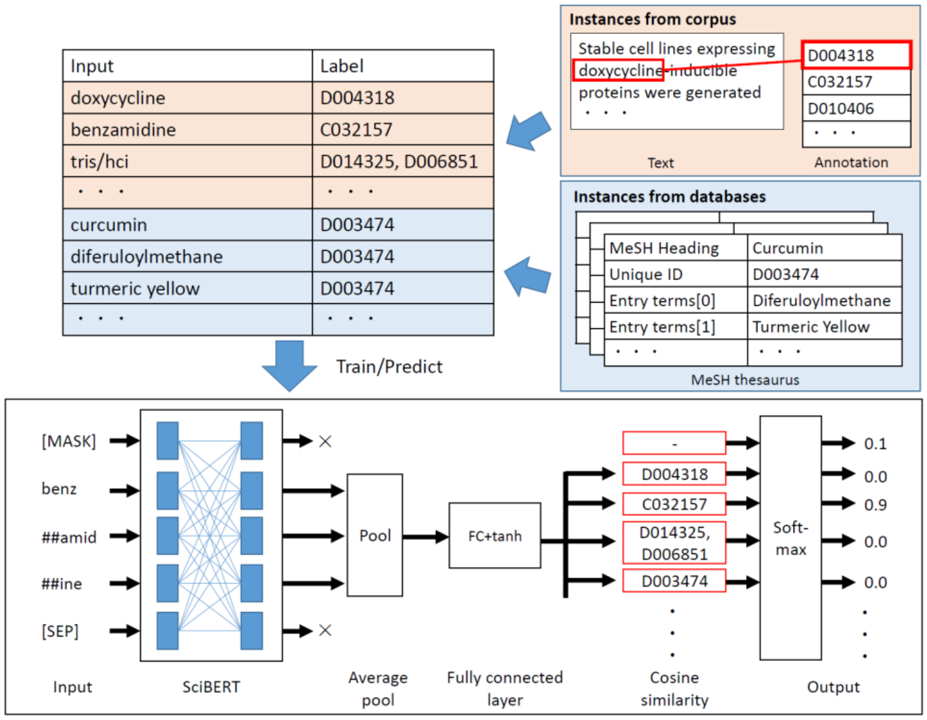

Fig. 1. The overview of the linking model.

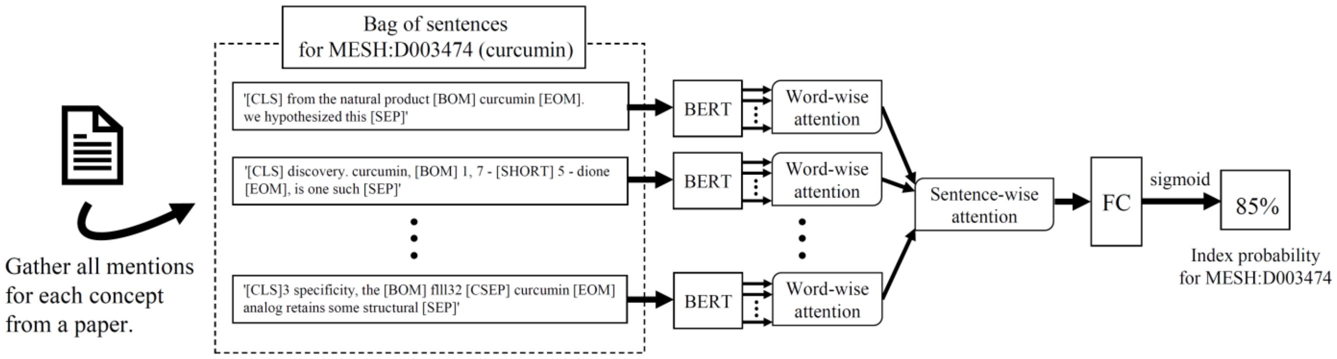
Bag of sentences for MESH:D003474 (curcumin)
'[CLS] from the natural product [BOM] curcumin [EOM]. we hypothesized this [SEP]'
'[CLS] discovery. curcumin, [BOM] 1, 7 - [SHORT] 5 - dione [EOM], is one such [SEP]'
'[CLS]3 specificity, the [BOM] fll32 [CSEP] curcumin [EOM] analog retains some structural [SEP]'
Gather all mentions for each concept from a paper.
BERT
Word-wise attention
Sentence-wise attention
FC
sigmoid
85%
Index probability for MESH:D003474

Fig. 2. The overview of the neural network based indexing model.