CN-HIT-IT.NLP at SemEval-2020 Task 4: Enhanced Language Representation with Multiple Knowledge Triples

Yice Zhang, Jiaxuan Lin, Yang Fan, Peng Jin, Yuanchao Liu, Bingquan Liu
Intelligence Technology and Natural Language Processing Lab
School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
{yczhang, linjiaxuan, yfan, pjin}@insun.hit.edu.cn
{ycliu, liubq}@hit.edu.cn

Abstract

This paper describes our system that participated in the SemEval-2020 task 4: Commonsense Validation and Explanation. For this task, it is obvious that external knowledge, such as knowledge graph, can help the model understand commonsense in natural language statements. But how to select the right triples for statements remains unsolved, so how to reduce the interference of irrelevant triples on model performance is a research focus. This paper adopt a modified K-BERT as the language encoder, to enhance language representation through triples from knowledge graphs. Experiments show that our method is better than models without external knowledge, and is slightly better than the original K-BERT. We got an accuracy score of 0.97 in subtaskA, ranking 1/45, and got an accuracy score of 0.948, ranking 2/35.

1 Introduction

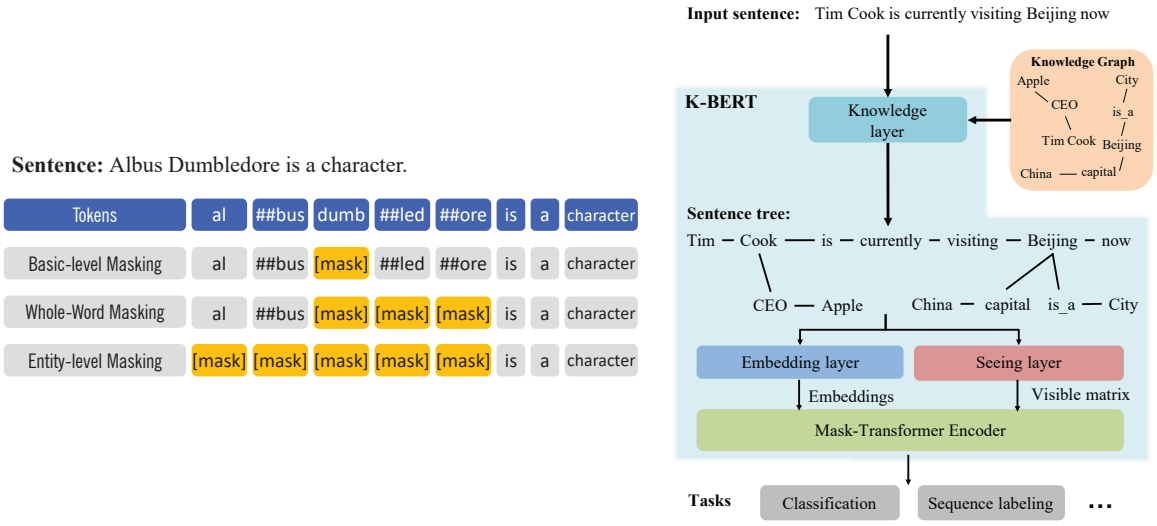
In recent years, language models trained on large scale corpora (Peters et al., 2018; Devlin et al., 2019; Lan et al., 2019) have performed exceptionally well on many benchmarks (Devlin et al., 2019), reaching or even surpassing human performance. It seems to show that Natural Language Understanding (NLU) is becoming easier. And the level of NLU may be seen in the ability to understand commonsense in natural language statements. Therefore, it is important to be able to evaluate how well a model can do for sense making(Wang et al., 2019).

For more direct research on commonsense in natural language statements, Commonsense Validation and Explanation (ComVE) is proposed by Wang et al.(2020). The ComVE task consists of three subtasks, and we participated in subtaskA and subtaskB. SubtaskA is validation, requiring the model to identify which statement makes sense from the given two. Then for the against-common-sense statement, three optional sentences are provided to explain why the statement does not make sense. In subtaskB, named Explanation (multi-choice), the only one correct reason is required to be identified from two other confusing ones.

Intuitively, knowledge graphs (KGs) can help language models understand commonsense. For example, for the sentence, "all whales are small", the triple <whale, hasproperty, big> are useful. Under such motivation, K-BERT (Liu et al., 2019) injects triples into sentences as domain knowledge. However, how to select the helpful triples from KG remains a problem. When we fuse too much external knowledge, irrelevant knowledge will adversely affect the model, which is called knowledge noise (KN) issue.

To overcome KN issue, K-BERT introduces soft-position and visible matrix to limit the influence of the triples. But KN issue is still serious, especially when the number of injected triples increases. Therefore, this paper introduce a variant of K-BERT for further reduction of KN. Besides, we choose ConceptNet (Speer et al., 2017) as the commonsense repository. Different from the domain-specific KG, ConceptNet contains more than 1,500,000 English nodes. This means that you can find almost every word in ConceptNet, which make it more difficult to choose the relevant triples. So this paper adopt a simple and junior threshold-based method to deal with this problem.

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 494–500
Barcelona, Spain (Online), December 12, 2020.



(a) Whole-word masking and entity-level masking.

(b) The model structure of K-BERT.

Figure 1: Some related work about pre-trained language models with knowledge integration.

2 Related Work

2.1 Pre-trained Language Models with Knowledge Integration

BERT-WWM (Cui et al., 2019) and Baidu-ERNIE (Sun et al., 2019) mask whole words or entire entities during the BERT's pre-training stage to introduce entity-level information. THU-ERNIE (Zhang et al., 2019) modify the encoder of BERT for the further mutual integration of words and entities. These works only utilize information at the entity level.

Kwon et al.(2019) proposes a method that integrates the triples from KG into texts. For an input text, it first extracts triples whose head and tail entities appear in the text. Then each triple is encoded as a single vector using an encoder, and these vectors are gathered to form a knowledge embedding. Finally, knowledge embedding is selectively fused into text representation obtained by BERT. However, this method may have Heterogeneous Embedding Space (HES) issue: the embedding vectors of words in text and entities in KG are obtained in separate ways, making their vector-space inconsistent. Moreover, in this method, triples don't have an impact on the encoding process of BERT.

To avoid HES issue, K-BERT (Liu et al., 2019) adopts a novel strategy to enhance language representation with triples. As shown in the figure 1(b), for an input sentence, K-BERT first injects relevant triples into it from a KG, producing a knowledge-rich sentence tree. Then the sentence tree is input into a mask-transformer, where a visible matrix is used to make the triples visible only to the corresponding entity. The difference with general transformer (Vaswani et al., 2017) is that mask-transformer uses a mask-self-attention instead, which can be illustrated by equation (1).

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^\top}{\sqrt{d_k}} + M)V, \tag{1}$$

where $Q, K \in \mathbb{R}^{n \times d_k}, V \in \mathbb{R}^{n \times d_v}$, and $M \in \mathbb{R}^{n \times n}$ denotes the visible matrix. The value in the visible matrix takes 0 or a large negative number, such as -10000. If w_j is invisible to w_i, M_{ij} will be set to -10000, which will mask the corresponding attention score to 0, meaning w_j make no contribution to the hidden state of w_i. In figure 1(b), for the entity *Beijing*, there two relevant triples, `<Beijing, capital, China>` and `<Beijing, is_a, City>`. In K-BERT, *capital* and *China* is only visible to *Beijing*, but *capital* and *is_a* are not mutually visible.

2.2 ConceptNet

In this paper, we choose ConceptNet[1] (Speer et al., 2017), an open multilingual knowledge graph, as the commonsense repository. ConceptNet contains approximately 34 million edges and over 8 million nodes. Its English vocabulary contains approximately 1,500,000 nodes, and there are 83 languages in which it contains at least 10,000 nodes. Currently, 34 relationships are defined in ConceptNet, such as `IsA`, `Synonym`, `PartOf`, `UsedFor` and so on.

3 System Overview

3.1 Notation

Given 5 sentences s_1, s_2, o_1, o_2, o_3, of which s_1, s_2 are two similar statements and only one statement makes sense; o_1, o_2, o_3 are optional sentences, and only one sentence can explain why the against-common-sense statement doesn't make sense. We denote common-sense statement as s_{ya}, the against-common-sense statement as $s_{\overline{ya}}$, and the corresponding reason as o_{yb}.

3.2 SubtaskA

In subtaskA, statement s first is encoded as $H \in \mathbb{R}^{d_1 \times |s|}$ by a encoder, which will be illustrated in section 4, and then through the attention mechanism, each token of H is merged to get $h \in \mathbb{R}^{d_1 \times 1}$.

$$H = \text{Encoder}(s), \tag{2}$$

$$h = \sum_t \alpha_t^a H_t, \tag{3}$$

where α_t^a is the attention score, which could be obtained by:

$$\alpha_t^a = \frac{\exp(k_t^\top q)}{\sum_t \exp(k_t^\top q)}, \tag{4}$$

$$k_t = \tanh(W_1 H_t + b_1), \tag{5}$$

where $W_1 \in \mathbb{R}^{d_2 \times d_1}, b_1, q_1 \in \mathbb{R}^{d_2 \times 1}$ are model parameters and d_2 is the attention size.

After the above encoding stage, we can get the statement vector h^{s_1}, h^{s_2} for the statements s_1, s_2. Next, as shown in equation (6), the class probabilities are calculated through the model parameters $W_2 \in \mathbb{R}^{1 \times d_1}, b_2 \in \mathbb{R}^{1 \times 1}$. Finally, we use the cross-entropy function to calculate the loss of subtaskA.

$$p_i = \frac{\exp(W_2 h^{s_i} + b_2)}{\sum_j \exp(W_2 h^{s_j} + b_2)}. \tag{6}$$

3.3 SubtaskB

In subtaskB, each sentence o will be concatenated together with s_{ya} and $s_{\overline{ya}}$. With the same encoder in section 3.2, we get $H^{s_{ya}} \in \mathbb{R}^{d_1 \times |s_{ya}|}, H^{s_{\overline{ya}}} \in \mathbb{R}^{d_1 \times |s_{\overline{ya}}|}, H^o \in \mathbb{R}^{d_1 \times |o|}$, and merge each token of H^o to get $h^o \in \mathbb{R}^{d_1 \times 1}$, using same attention mechanism as equation (4,5).

$$H^{s_{ya}}, H^{s_{\overline{ya}}}, H^o = \text{Encoder}(s_{ya}, s_{\overline{ya}}, o), \tag{7}$$

$$h^o = \sum_t \alpha_t^b H_t^o. \tag{8}$$

Note that only h^o will be used in the subsequent classification, and $H^{s_{ya}}$ and $H^{s_{\overline{ya}}}$ will be discarded. This means that s_1 and s_2 are only used to enhance the representation of sentence o during the encoding stage. Next, the class probabilities are calculated similarly as equation (6) and the loss is calculated by the cross-entropy function.

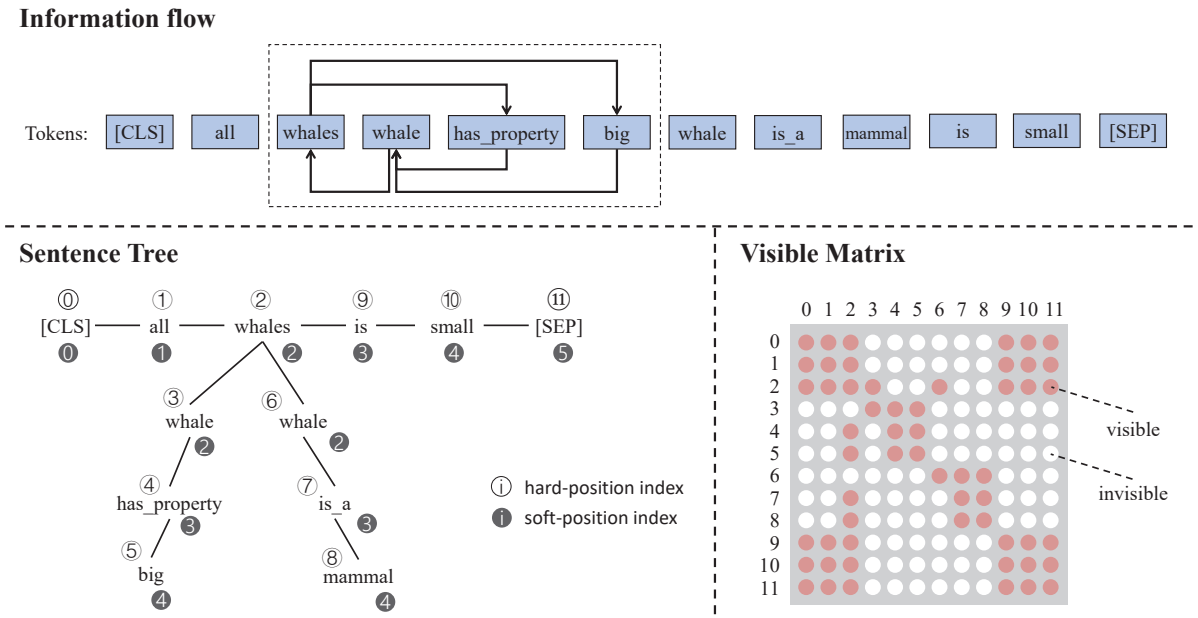


Figure 2: A simple examples for our sentence tree, visible matrix and information flow. In the visible matrix, if M_{ij} is invisible, w_j will make no contribution to the hidden state of w_i.

4 Encoder Enhanced by Triples

In this section, we describe our encoder, which is a variant of K-BERT (Liu et al., 2019). Its general framework is the same as K-BERT presented in Figure 1(b), but some details are modified.

4.1 Knowledge Layer

For an input sentence, the knowledge layer first recognizes entities in it according to a threshold-based strategy, and then these entities will be sent to a KG to get relevant triples. With these triples, original sentence will be transformed into a knowledge-rich sentence tree.

Let's denote the input sentence as $s = \{w_1, w_2, \cdots, w_{|s|}\}$. If the word frequency of the word w_i, denoted as $f(w_i)$, is less than a given threshold θ, w_i will be regarded as a entity. Therefore, the entities we recognize are all single words. Then each recognized entity is input into the KG $\mathbb{K}$ as a query to obtain the corresponding triples. Because the knowledge graph we choose is too large, a query is only performed on a given relation. Hence, the query process can be formalized as:

$$E = query(e, \mathbb{K}, rel). \tag{9}$$

After recognizing the entities and querying the triples, for the sentence s, we get $\mathbb{E} = \{(w_i, E_i), (w_j, E_j), \cdots\}$, where $E_i = \{(w_i, r_{i1}, w_{i1}), \cdots, (w_i, r_{ik}, w_{ik})\}$. As shown in figure 2, triples in $\mathbb{E}$ are stitched in the corresponding position in s, producing a sentence tree. This process can be formulated as equation (10). Note that the head entity is preserved when stitching, which is different from the original K-BERT.

$$T = inject(s, \mathbb{E}). \tag{10}$$

4.2 Soft-Position Embedding

Since the self-attention mechanism cannot make use of the order of the sequence, the transformer (Vaswani et al., 2017) adds a position embedding to the input embedding, named hard-position embedding in this paper. Similarly, K-BERT introduces soft-position embedding to preserve the structure information of the sentence tree. As shown in the figure 2, the hard-position index of token *has_property*

[1]we use ConceptNet 5.7.0 version, which could be downloaded at: `https://s3.amazonaws.com/conceptnet/downloads/2019/edges/conceptnet-assertions-5.7.0.csv.gz`

Subtask	Train	Dev	Test
A	10770	997	1000
B	10808	997	1000

Table 1: Data statistics.

and token *is* are 3 and 9 respectively, but their soft-position index are 2 and 3. This means that with soft-position, the injection of triples doesn't affect the position index of tokens in the original sentence.

4.3 Mask-Self-Attention and Visible Matrix

The basic idea of mask-self-attention can be illustrated by equation (1), namely using a visible matrix $M \in \mathbb{R}^{n \times n}$ to control the information flow, where n is the length of the input sequence.

There are some differences in the visible matrix between our encoder and the original K-BERT. For the given entity e in the sentence and the corresponding triple $< h, r, t >$, the information flow in the following three directions is allowed:

1. from the entity e to the relation r and the tail entity t of the triple;
2. from the relation r and the tail entity t to the head entity h of the triple;
3. from head entity h to the entity e.

Unlike in K-BERT, the direct information flow from the relation r and the tail entity t to the entity e is masked. This means that in the triple $< h, r, t >$, the entity e can only be affected by the head entity h, which could be understood as a representation of the entity e enhanced by the relation r and the tail entity t.

5 Experiment Results

5.1 Experiment Setup

Data. We merge the given train data and the given trial data into the new train data, and remove duplicate samples and false samples[2]. Table 1 shows the statistics of the data used.

Model and Hyper-parameters. The implementation of our encoder is based on Google ALBERT[3] (Lan et al., 2019). In ALBERT we used, the hidden size d_1 is equal to 4096, and we set the attention size d_2 as 1024. Our model is trained for 4 epochs with a initial learning rate[4] of 1e-5 and the batch size of 16.

Other Details. We used a python library named `wordfreq` (Speer et al., 2018) to look up the word frequency. It can output the zipf frequency of words, ranging from 0 to 8. And we empirically set the frequency threshold θ as 4. For ConceptNet, we select 10 relations, as shown in table 2. Among the 34 relations of ConceptNet, we filter out relations that contain little commonsense, such as `RelatedTo`, and relations with too few triples that cannot involve enough samples, such as `CreatedBy`. For a given entity and a given relationship, we choose the k triples with the highest confidence, which is provided by ConceptNet. k ranges from 1 to 4, and the best k of each relation is determined on the development set in subtaskA.

5.2 Results

As shown in table 2, on the test set of subtaskA, the accuracy score of our method is best to reach 0.9670, which exceeds 0.9600 obtained by naive ALBERT. In terms of the average accuracy score on 10 relations, our method is slightly better than K-BERT. This shows that the injection of triples is helpful

[2]In some samples, s_1 and s_2 are the same statement.

[3]We initialize our model with albert-xxlarge-v2.

[4]We apply the `cosine_with_hard_restarts_schedule_with_warmup` with warmup proportion of 0.2 to automatically adjust the learning rate.

Relation	Dev		Test		Avg_1
	K-BERT	Ours	K-BERT	Ours	
IsA	0.9669	0.9669	0.9620	**0.9630**	0.9647
Synonym	0.9649	**0.9679**	0.9670	0.9670	0.9667
HasContext	0.9699	**0.9709**	0.9630	**0.9650**	0.9672
CapableOf	**0.9669**	0.9659	0.9620	**0.9660**	0.9652
AtLocation	0.9669	0.9669	**0.9630**	0.9620	0.9647
UsedFor	0.9659	**0.9679**	0.9600	**0.9640**	0.9645
HasProperty	0.9639	**0.9679**	**0.9650**	0.9640	0.9653
ReceivesAction	0.9629	**0.9659**	0.9630	**0.9660**	0.9645
HasA	0.9649	**0.9659**	**0.9670**	0.9660	0.9660
PartOf	0.9629	**0.9669**	**0.9670**	0.9650	0.9655
Avg_2	0.9656	**0.9673**	0.9639	**0.9648**	0.9654
naive-albert	0.9649		0.9600		0.9625

Table 2: Result of classification accuracy for different ConceptNet relations on subtaskA development set and test set. Naive-albert means that the triples in conceptnet is not used. Avg_1 refers to the average accuracy of K-BERT and our method on the development set and test set. Avg_2 refers to the average accuracy on all relations.

for the understanding of commonsense in natural language statement, and our method may be better than the original K-BERT for knowledge integration.

`HasContext` and `Synonym` are the best two relations for subtaskA. As shown in four statements below, this may be because triples of these relation provide some optional context to enhanced the entity representation or reduce the difficulty of the model in understanding low-frequency words.

> Statement1(False): "the family adopted a dinosaur (dinosaur, hascontext, proscribed) to be their new pet".
>
> Statement2(True): "the mclaren (mclaren, hascontext, automotive) is a well-designed vehicle.".
>
> Statement3(False): "robot be such omniscient (omniscient, synonym, all knowing) as human".
>
> Statement4(False): "cat be bipedal (bipedal, synonym, two footed) creature".

Note that in the official submission, we train our model in the train data and the development data together, and adopt 5-fold cross-validation and soft-vote strategy to obtain the final probability distribution on the test set. Besides, only two types of relations, `IsA` and `Synonym`, are used. Under such setting, we achieved an accuracy score of 0.970 on subtaskA, ranking 1/45, and achieved an accuracy score of 0.948 on subtaskB, ranking 2/35.

6 Conclusion

In this paper, we propose a variant of K-BERT as the language encoder to help the model understand commonsense. We adopt a simple threshold-based method for triple selection and apply our encoder on SemEval-2020 task 4: Commonsense Validation and Explanation, and achieve good performance (ranking 1st place in subtskA). This shows that our system can effectively enhance the language representation with multiple knowledge triples.

However, when the number of injected triples is further increased, or triples of different relations are injected into the sentence, the performance of our model will deteriorate. Therefore, we believe that future work can be carried out in the following points: (1) pre-training language models on sentences injected with triples; (2) recognizing multi-word entities an d developing better selection methods, such as those based on similarity; (3) designing a robust and noise-resistant knowledge integration model to integrate more triples.

Acknowledgements

This study was supported by the National Natural Science Foundation of China (No. 61672192).

References

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186.

Sunjae Kwon, Cheongwoong Kang, Jiyeon Han, and Jaesik Choi. 2019. Why do masked neural language models still need common sense knowledge?

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2019. K-bert: Enabling language representation with knowledge graph. *arXiv preprint arXiv:1909.07606*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Robyn Speer, Joshua Chin, Andrew Lin, Sara Jewett, and Lance Nathan. 2018. Luminosoinsight/wordfreq: v2.2, October.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. 2019. Does it make sense? and why? a pilot study for sense making and explanation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4020–4026, Florence, Italy, July. Association for Computational Linguistics.

Cunxiang Wang, Shuailong Liang, Yili Jin, Yilong Wang, Xiaodan Zhu, and Yue Zhang. 2020. SemEval-2020 task 4: Commonsense validation and explanation. In *Proceedings of The 14th International Workshop on Semantic Evaluation*. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451.