Automated Math Word Problem Knowledge Component Labeling and Recommendation

Chor Seng Tan(✉) and Jung-Jae Kim

Institute for Infocomm Research (I2R), A*STAR, Singapore, Singapore
{tan_chor_seng,jjkim}@i2r.a-star.edu.sg
https://www.a-star.edu.sg/i2r

Abstract. The accurate annotation of math exercises and word problems according to a diverse set of knowledge components is an important task for many education applications. It is an extremely complex and resource-intensive process, and has traditionally been done manually by experienced educators. There has been research work in recent years to apply machine learning to automate the process, however, due to the varied datasets used by different researchers and the private nature of these datasets, there is no good benchmark for which model performs the best for such a task. Moreover, the datasets used in literature typically comprise math exercises that follow similar templates. In this paper, we benchmark some of the best reported models on a math word problem dataset with fine-grained knowledge component annotations, and highlight the challenges in making accurate predictions. We propose models to improve the prediction accuracy, and demonstrate how they can be used to extract similar problems based on knowledge component from an unlabelled pool of questions, thus empowering educators to better identify the knowledge gaps of their students and tailor suitable practice problems for them.

Keywords: Math Word Problems · Knowledge component · Annotation · Recommendation · BERT

1 Introduction

Accurate annotation of math exercises and problems with relevant knowledge components from the curriculum is extremely important to online tutoring platforms such as ASSISTments and Cognitive Tutors for the purposes of personalized recommendation [1,2] and knowledge tracing [3,4] to help tailor the best learning experience for students. The task of annotating new math questions is laborious as the range of knowledge components can go up to the hundreds. For instance, the Common Core State Standards used in the US Curriculum has over 200 knowledge components for K-12 mathematics [5]. It is typically performed by experienced domain experts, usually teachers, but even then, there can often

C. Herodotou et al. (Eds.): MIS4TEL 2024, LNNS 1171, pp. 1–11, 2024.
https://doi.org/10.1007/978-3-031-73538-7_30

be disagreements with the labeling due to how challenging it is [6], thus more cross-checks are required to ensure consistency, resulting in even more work.

In recent years, many researchers have proposed applying machine learning methods to solve this problem using both unsupervised learning [7] and supervised learning techniques [8,9], with some promising results. Usually this task is framed as a multi-class classification problem [9] where the content of a math problem is analyzed by a model, which predicts the correct knowledge component for it. However, these works train and evaluate their models on different data sources that are not publicly released, making them difficult to benchmark and analyze. Moreover, there have been reports that the math exercises used in these works follow certain templates that lead the model to overfit to the templates, thus being non-generalizable to different data sets [6]. In this paper, we benchmark different models on challenging math word problem datasets with fine-grained knowledge component annotations that do not follow fixed question templates, to better compare the models. We also highlight some challenges with knowledge component annotation of math questions for practical usages and propose some solutions to address them.

2 Related Work

Early work on automated knowledge component annotation operated on the assumption that problems with the same tags share regular keywords [8], leading to models that learn to label math problems based on the similarity of their word distributions to tagged content. One of the earliest reported work was by Cetintas et al. [10], who trained a support vector machine (SVM) classifier to differentiate Multiplicative Compare and Equal Group problems from other types of math problems from a fourth grade math textbook. Later, Karlovcec et al. [11] employed a text mining approach with SVM and K-nearest neighbour on mathematical text with up to 106 knowledge components taken from the ASSISTments Platform between 2005 to 2006, and they were able to identify the correct tag in 62.1% of cases. This value goes up to 88.9% when 10 suggestions are provided. Working on ASSISTment problems from 2012–2023, Pardos et al. [8] presented a model that can accurately match problems with their associated knowledge components out of a pool of 198 in total, achieving an accuracy of 90%, using a combination of skip-gram and bag-of-words.

More recently, Shen et al. [9,12] demonstrated improved performance with models based on BERT (Bidirectional Encoder Representations from Transformers) [13]. They introduced Task-Adaptive Pretrained (TAPT) BERT, that is pretrained on various math data, as well as MathBERT, a model based on BERT that is further pretrained on a large mathematical corpus ranging from prekindergarden all the way to college level. Tested on an ASSISTments dataset with 13722 problems and 213 possible labels, MathBERT was able to achieve an F1 score as high as 92.67% and an accuracy as high as 93.92%. There have also been other works on knowledge component labelling for different languages such as Japanese [14] and Chinese [15], but they use relatively small and inaccessible datasets.

There has been some controversy over the dataset used in some of these works. While replicating the methodology used by Pardos et al., Patikorn et al. identified an issue with using certified Skill Builder problems from ASSISTments as many of the questions are generated based on templates and are nearly identical with only differing numerical values in most cases [6]. Patikorn purported that this caused models to remember specific words in the templates instead of learning the terminology of the skills, inflating the prediction performance. As evidence, Patikorn replicated the results of Pardo et al. on ASSISTments data, but showed that the performance dropped drastically, from over 90% to less than 15% when tested on a dataset from Illustrative Mathematics, a different data source also based on the Common Core standard. This showed that the models developed lacked generalizability and were not ready for real-world application. Additionally, some of the works report a reliance on key phrases in the math exercises to make predictions, suggesting that their models do not actually understand the underlying math concepts in the problems [14].

In this work we conduct experiments to answer the following questions:

1. Do the models and methods reported by recent literature and the latest large language models (LLMs) perform well on more complex math word problem datasets that contain more realistic problems which do not follow templates?
2. What are some of the key features that affect model accuracy and how can the automated knowledge component annotation be further improved?
3. Can we leverage on these models to make recommendations of similar math word problems based on the knowledge component?

3 Methodology

3.1 Data

We use two publicly-available math word problem datasets in English language that were introduced by TAL Education Group for the purpose of training and evaluating math word problem solvers. The first dataset is TAL-SCQ5K-EN, a high quality mathematical competition dataset comprising a total of 5000 questions that span topics from primary, junior high, and high school levels. The other dataset, TAL-SAQ6K-EN, was later introduced as the test set for the AAAI2024 Global Challenge on Math Word Problem Solving, and consists of 5927 questions. Questions from both datasets are annotated with detailed knowledge point route(s) that can go up to 5 levels, thus presenting valuable fine-grained annotations for model development and evaluation. All the mathematical expressions in the datasets are presented in standard text-mode Latex.

To evaluate knowledge component annotation models, we use word problems from TAL-SCQ5K-EN as the training set and those from TAL-SAQ6K-EN as the evaluation set, and use the knowledge point routes as knowledge components. We remove the problems with multiple knowledge components to train and evaluate our model on only a single knowledge component per problem. To ensure that the annotations are fine-grained, we remove problems with knowledge components

that have only one or two levels. As majority of the problems' knowledge components do not extend beyond three levels, we terminate the longer ones at level 3. From the training set, we also remove the knowledge components with fewer than 5 examples, as there will be too few examples for the model to learn from. We end up with 2247 problems with 109 unique knowledge components in the training set. For the evaluation set, we have 3234 problems with 206 knowledge components, of which 103 were seen in the training set and 103 are completely unseen. This deviation from more conventional 70:20 or 80:20 train:test splits is intentional as we want the evaluation set to be as vast as possible relative to the training set to better resemble realistic scenarios. We keep the questions with unseen knowledge components in the evaluation set as it is common to encounter unseen labels in real-life application, and we will compare the model's confidence on seen versus unseen labels in the Results section. Overall, the evaluation questions that have knowledge components found in the training data comprise 78.8% of the evaluation set.

To demonstrate that the math questions in our dataset are truly diverse, we employ sentence vectorization to obtain the sentence embeddings and compare them using cosine similarity. We use two different well-established models for vectorization, namely bag-of-words vectorization [16] and Sentence-BERT [17] vectorization and compare the similarities between math word problems of the same and different knowledge components for both the train and evaluation datasets. The cosine similarities obtained are summarized in Table 1. We observe negligible difference between the sentence embeddings for both datasets across the knowledge components with both models, confirming that the TAL dataset word problems are indeed well-diversified and do not follow templates.

Table 1. Cosine similarity between the sentence embeddings

	Train set cosine similarity		Evaluation set cosine similarity	
	Same KC	Different KC	Same KC	Different KC
Bag-of-words	0.1591	0.1592	0.1771	0.1762
Sentence-BERT	0.2237	0.2237	0.1657	0.1642

3.2 Models

Benchmarking. For benchmarking purposes, we pose the knowledge component classification task as a multi-class classification problem, in line with the recently reported literature on this topic [9,12]. Models are given the problem text as input and trained to predict the correct knowledge component out of all possible 'seen' knowledge components. We benchmark the following models/methods:

1. Fasttext vectorization of problem text [18] + XGBoost classification model

2. Multi-class classification with fine-tuned open-source large language models. Besides MathBERT, we also compare with BERT-base, BERT-large, and RoBERTa-large [19] to study the effect of different model sizes and pretraining. We train all models with a batch size of 16 and a learning rate of 1e-5 on an NVIDIA A40 GPU.

Since being introduced to the public in November 2022, ChatGPT [20] has become widely used for personal as well as commercial use for tasks such as question and answering and code generation. To study its usefulness for knowledge component annotations without any finetuning, we use the GPT-3.5 Turbo model from OpenAI, and prompt it with the following system role: *"You are an experienced math educator, and highly skilled with annotating math word problems with the correct knowledge label from a pool of candidates."*, this is important to set the behavior of the model for our desired task. Next we prompt the model to pick the right knowledge component label for each question as follows: *"What is the correct label for the question: <question>, please select the most suitable knowledge point route from the following options: <all kcs>"*, where *<question>* is the text for each question and *<all kcs>* represents a string of all the knowledge point routes in the training set, separated by a comma. The option selected by the model is recorded and compared to the ground truth.

4 Results

4.1 Benchmarking

To obtain the benchmarking results, we perform the training of each model 5 times and the results on the test set are averaged and shown in Table 2. As we have over 100 class labels, we report top-N accuracy, which measures the accuracy of the true class being within the N most probable classes predicted by the model. We use values of 1, 3, 5, and 10 for N.

We see that the results for Fasttext + XGBoost method performs the worst, with a top-1 accuracy of only 0.124. Meanwhile, the BERT-based models show much improved performance, with a top-1 accuracy ranging from 0.309 for BERT-large to 0.359 for RoBERTa-large. For top-3, top-5, and top-10 accuracy, we also observe that the best results are obtained by RoBERTa-large, while MathBERT, BERT-base, and BERT-large perform very similarly. These results suggest that neither the larger parameter size of BERT-large over BERT-base, nor the pretraining on mathematical corpus of MathBERT over BERT-base led to much improvement. Instead, more optimized training methods in the case of RoBERTa-large, which was trained for longer, with larger batch sizes and longer sequences of data, and used dynamic masking compared to the BERT models, can yield better performing models for math problem annotations [19].

We can see that when applied to the challenging TAL dataset, there is a significant drop in prediction accuracy by models that reported over 90% accuracy on their own private datasets, suggesting that those results could be due to the ease of those datasets and possible overfitting to repeated key phrases [8,12]. One

reason for the relatively lower accuracy scores on our evaluation dataset is the presence of unseen labels, which comprise about 21.2% of the data. If we consider only the accuracy metrics on problems with knowledge point routes that exist in the training set, we can get a top-1 accuracy of 0.455 and top-10 accuracy of 0.815 for the best-performing RoBERTa-large model. The "in-train"-only results are summarized in the right columns of Table 2.

Table 2. Benchmarking results

Model	Accuracy				Accuracy (in-train only)			
	Top-1	Top-3	Top-5	Top-10	Top-1	Top-3	Top-5	Top-10
Fasttext + XGB	0.124	–	–	–	0.158	–	–	–
Multi-class (MathBERT)	0.323	0.479	0.535	0.597	0.409	0.608	0.6789	0.757
Multi-class (BERT-base)	0.327	0.477	0.531	0.600	0.415	0.605	0.673	0.761
Multi-class (BERT-large)	0.309	0.471	0.530	0.602	0.391	0.598	0.672	0.763
Multi-class (RoBERTa-large)	0.359	0.535	0.589	0.642	0.455	0.678	0.747	0.815

For knowledge component annotation using GPT-3.5 Turbo, we obtained an overall top-1 accuracy of 0.053, which is worse than even the Fasttext + XGB method. The low performance suggests that the terminology and complexity of the knowledge components of the TAL datasets is very different from the data exposed to the commercial LLMs during pretraining, and using these commercial LLMs off-the-shelf is insufficient without performing further finetuning on dedicated task data.

4.2 Analysis

How Many Training Examples Are Required for Each Unseen Label? As manual annotation of training examples is costly, we should determine the minimum number of training examples per unseen label to achieve good performance. For this purpose, we examine how the prediction accuracy for each label class for the top-performing RoBERTa multi-class classification model varies with the number of examples present in the training set. We plot scatter plots for all the top-N accuracies in Fig. 1. Across the different values for N, we observe that at small training examples (less than 30), there is a wide spread in the accuracy, spanning from 0 to 1.0. This reveals that while some knowledge components can achieve high accuracy with relatively few examples, others perform poorly. As the number of training examples increases, we observe that the accuracy stabilizes. In short, there is a tendency that more training examples lead to higher annotation performance, up till a certain point beyond which the performance plateaus. For the TAL dataset, this occurs at around 50 training examples.

Can Machine Learning Models Produce Reliable Confidence Scores of Predicted Knowledge Components for Individual Questions? Besides simply looking at the top-N predictions provided by a language model, we can also consider the confidence of each prediction, which is typically calculated by the difference between the softmax prediction probability of the top two candidates. Figure 2(a) plots the top-1 accuracy of the RoBERTa-large classification model vs. the confidence of the prediction, and we can observe that the accuracy generally increases with prediction confidence. This means that by setting a confidence threshold for predictions, we can ensure that the model only makes annotation suggestions for high confidence predictions with correspondingly higher accuracy. For instance, at 98% confidence and above, the top-1 accuracy is 0.673, a large improvement over the overall accuracy of 0.366. We further investigate the model's confidence for problems with knowledge components that are found in the training set versus those that are not, and found that the mean prediction confidence for the former is 72.1% compared to 58.8% for the latter, indicating that the model is more confident about the predictions for previously-seen knowledge components.

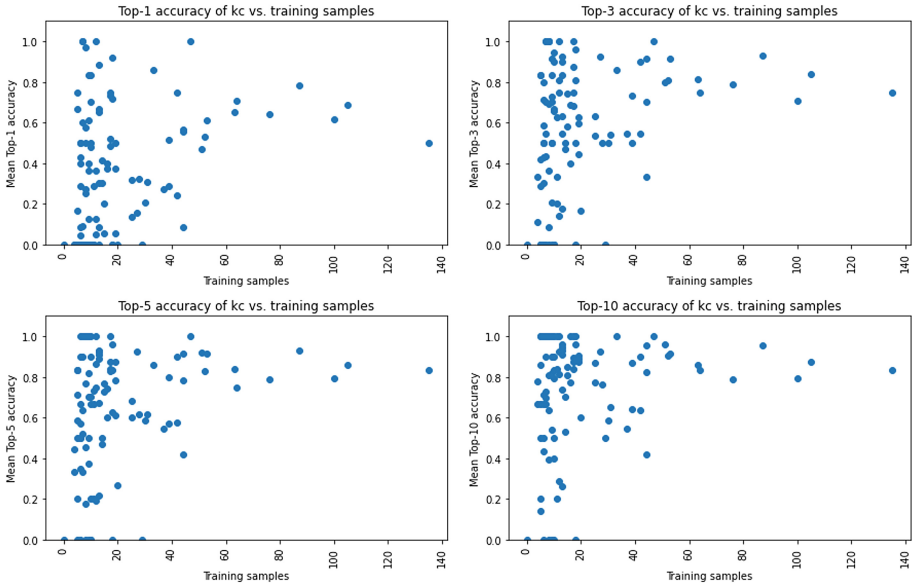
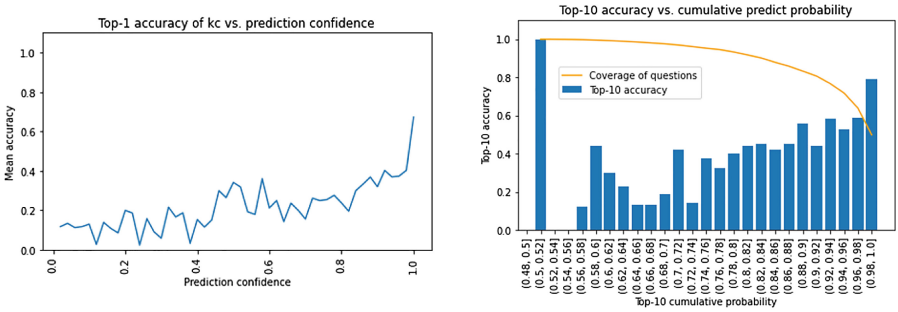


Fig. 1. Number of training examples vs. accuracy

We also study the relationship between the top-10 accuracy and the prediction probabilities, using the cumulative probabilities of the top 10 classes as a measure of confidence. In Fig. 2(b) we divide the cumulative probabilities into bins of width 0.02 and plot the top-10 accuracy against it. We also plot the coverage of the problems above a certain cumulative probability, denoted by the



(a) Top-1 accuracy vs. confidence (b) Top-10 accuracy vs. cum probability

Fig. 2. Plots for prediction accuracy vs. prediction confidence and cumulative probability

orange curve in the figure. We observe a similar trend of accuracy increasing with cumulative probability. There is a strange peak with accuracy of 1.0 for the confidence bin between 0.50 and 0.52, but there is only one problem in that bin so it is disregarded as an anomaly. In the highest bin with cumulative probability above 0.98, the accuracy is 0.79 with about 50% of the problems, indicating that by setting a high cumulative probability threshold, we can filter for the higher accuracy predictions whilst maintaining a good coverage of all questions. AQ4

4.3 Problem Recommendation Based on Similar Knowledge Components

A key benefit of having math problems annotated with knowledge components is the ability to recommend other questions with the same knowledge components. One way to do this is by considering the class (knowledge component) probabilities for each question and finding other questions with the most similar probabilities. In this section we investigate how many of the other questions of the same knowledge component are retrieved (or recommended) for the given question with high ranks. For this analysis, we only consider the problems with ground truth labels that were in the training set. For each math problem in the evaluation dataset, we computed and ranked the cosine similarity of its knowledge component class probabilities as predicted using RoBERTa-large against all other questions. We tabulated the mean cosine similarity of problems with the same knowledge component and found it to be 0.458, significantly higher than the overall mean of 0.027, proving that problems with the same knowledge component have more similar class probabilities.

To evaluate how well this approach performs for making problem recommendations we use hits@k, which considers the top k recommendations for each problem based on similarity and tabulates the number that have the same knowledge component. We use k = 1, 3, 5, and 10, and compare the hits@k for all the knowledge components as well as for only the knowledge components with

at least 30 and 50 training examples. The hit@1 was 0.53, 0.61, and 0.66 respectively, demonstrating that using cosine similarity of prediction probabilities is an effective way to retrieve similar problems. We also observe improvements in hits@3, hits@5, and hits@10 with more training examples, proving that more training examples is beneficial to making better problem recommendations. The results are consolidated in Table 3.

We compare the results to using Sentence-BERT to obtain the sentence embeddings of the question texts and computing the cosine similarity based on that to make recommendations of similar problems. Interestingly, the hits@1 for Sentence-BERT is slightly better than using the RoBERTa-large prediction probabilities. A possible reason for this is that for each knowledge component there are problems that are semantically very similar, so finding the one with the highest similarity yields good results. As k increases, we observe that the RoBERTa-large model performs significantly better, for example, hits@10 for RoBERTa-large with at least 50 training examples is 5.99 versus. 5.03 for Sentence-BERT. This shows that for making recommendations of a larger number of similar problems, finetuning a RoBERTa-large model produces superior results.

Table 3. Hits@k versus number of training examples

Hits@k	RoBERTa-large			Sentence-BERT		
	All	Min. 30 examples	Min. 50 examples	All	Min 30. examples	Min 50. examples
hits@1	0.53	0.61	0.66	0.57	0.62	0.67
hits@3	1.43	1.60	1.88	1.46	1.64	1.84
hits@5	2.28	2.69	3.04	2.18	2.48	2.83
hits@10	4.3	5.23	5.99	3.76	4.25	5.03

5 Conclusion

From our results, we conclude that accurately labeling the knowledge components of math word problems is a challenging task that even commercial LLMs such as GPT-3.5 Turbo are unable to solve without finetuning. We believe that human input is still necessary for annotation tasks, but finetuned models such as RoBERTa-large can aid in the process by providing suggestions for a manual annotator to select from. We show that with confidence thresholding, it is possible to filter for more accurate predictions, and also show that limitations on training data still constrain model performance at the top-1 accuracy level. It is possible to design an LLM-powered annotation system which makes suggestions for high-confidence predictions, which can be verified by human annotators, and added to the training data pool to improve the model further.

Acknowledgements. This research is supported by the Ministry of Education, Singapore, under its Science of Learning Grant (award ID: MOE-MOESOL2021-0006).

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore.

References

1. Hou, Y., et al.: Course recommendation of MOOC with big data support: a contextual online learning approach. In: IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE (2018)
2. Lin, Y., et al.: Adaptive course recommendation in MOOCs. Knowl.-Based Syst. **224**, 107085 (2021)
3. Piech, C., et al.: Deep knowledge tracing. In: Advances in Neural Information Processing Systems, vol. 28 (2015)
4. Vie, J.-J., Kashima, H.: Knowledge tracing machines: Factorization machines for knowledge tracing. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01 (2019)
5. Common Core Standards. https://corestandards.org/. Accessed April 2024
6. Patikorn, T., Deisadze, D., Grande, L., Yu, Z., Heffernan, N.: Generalizability of methods for imputing mathematical skills needed to solve problems from texts. In: Isotani, S., Millán, E., Ogan, A., Hastings, P., McLaren, B., Luckin, R. (eds.) AIED 2019. LNCS (LNAI), vol. 11625, pp. 396–405. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23204-7_33
7. Desmarais, M.C.: Conditions for effectively deriving a Q-Matrix from data with non-negative matrix factorization. In: Proceedings of Educational Data Mining (2011)
8. Pardos, Z.A., Dabu, A.: Imputing KCs with representations of problem content and context. In: Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization (2017)
9. Shen, J.T., et al.: Classifying math KCs via task-adaptive pre-trained BERT. Artif. Intell. Educ. (2021)
10. Cetintas, S., et al.: Automatic text categorization of mathematical word problems. In: Proceedings of the 22nd International FLAIRS Conference (2009)
11. Karlovcec, M., et al.: Knowledge component suggestion for untagged content in an intelligent tutoring system. In: Proceedings of the 11th International Conference on Intelligent Tutoring Systems (2012)
12. Shen, J.T., et al.: MathBERT: a pre-trained language model for general NLP tasks in mathematics education (2021)
13. Devlin, J., et al.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint(2018)
14. Yamauchi, T., et al.: Automated labelling of PDF mathematical exercises with word N-grams VSM classification. Smart Learn. Environ. (2023)
15. Ning, Z., Fan, O.: An integrated approach for knowledge extraction and analysis in collaborative knowledge construction. Int. J. Educ. Technol. High. Educ. (2023)
16. Qader, W.A., et al.: An overview of bag of words; importance, implementation, applications, and challenges. In: International Engineering Conference. IEEE (2019)
17. Reimers, N., Gurevych, I.: Sentence-BERT: sentence embeddings using Siamese BERT-networks. arXiv preprint arXiv:1908.10084 (2019)

18. Joulin, A., et al.: Bag of tricks for efficient text classification. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, pp. 427–431 (2017)
19. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint (2019)
20. OpenAI ChatGPT3.5. https://chat.openai.com/. Accessed April 2024

Author Queries

Chapter 30

Query Refs.	Details Required	Author's response
AQ1	This is to inform you that corresponding authors have been identified as per the information available in the Copyright form.	
AQ2	Please check and confirm if the authors Given and Family names have been correctly identified.	
AQ3	As Per Springer style, both city and country names must be present in the affiliations. Accordingly, we have inserted the city name in the affiliation. Please check and confirm if the inserted city name is correct. If not, please provide us with the correct city name.	
AQ4	To maintain sequential order, figures citations have been renumbered. Please check and correct if necessary.	