# Controlled Cloze-test Question Generation with Surrogate Models for IRT Assessment

**Anonymous ACL submission**

## Abstract

Item difficulty plays a crucial role in adaptive testing. However, few works have focused on generating questions of varying difficulty levels, especially for multiple-choice (MC) cloze tests. We propose training pre-trained language models (PLMs) as surrogate models to enable item response theory (IRT) assessment, avoiding the need for human test subjects. We also propose two strategies to control the difficulty levels of both the gaps and the distractors using ranking rules to reduce invalid distractors. Experimentation on a benchmark dataset demonstrates that our proposed framework and methods can effectively control and evaluate the difficulty levels of MC cloze tests.

## 1 Introduction

Multiple-choice cloze tests are fill-in-the-blank questions that assess reading comprehension and overall language proficiency by requiring test takers to select the correct missing words from options. Table 1 gives an example test item consisting of a stem with a gap to fill, a key or answer, and three distractors.

| **Stem:** |
|---|
| I knelt and put my arms around the child. Then the tears came, slowly at first , but soon she was ___ her heart out against my shoulder. |
| **Options:** |
| A. crying B. shouting C. drawing D. knocking |
| **Key:** A **Distractors:** B C D |

Table 1: A question item of MC Cloze test.

MC cloze test questions have been a focus of research because they are a common question format on standardized language proficiency exams such as TOEFL, TOEIC, IELTS, and college/high school entrance exams. In this paper, we address the research questions of generating MC cloze test of different item difficulty levels.

Prior studies on cloze test question generation have concentrated largely on distractor generation, with the goal of reproducing distractors exactly matching the benchmark datasets (Chung et al. 2020; Ren et al., 2021; Chiang et al. 2022; Wang et al. 2023). Although some studies have acknowledged the benefit of having distractors with diverse difficulty levels (Yeung et al., 2019), there has been minimal investigation into generating distractors with difficulty level different from the benchmark.

Item difficulty plays a crucial role in adaptive testing. It is a parameter that determines which questions to present to a test taker and estimates their proficiency level. Therefore, the difficulty of each item should be known beforehand so appropriate questions can be selected during the test (Susanti et al. 2017). However, only a number of works have focused on generating question items of various difficulty levels, for RC questions (Gao et al. 2019a), C-test questions (Lee et al. 2019 and 2020), and MC cloze questions (Susanti et al. 2017). This research gap is largely due to the lack of a reliable metric to evaluate the item difficulty of generated questions. Most previous study relies on human test takers and human annotation for assessing the change of difficulty levels (Susantia et al. 2017, Lee et al. 2019).

Our research has two main goals: (1) We propose strategies to generate cloze-test questions by controlling both the distractors and the gap, with consideration for reducing invalid distractors. (2) We address the problem of objective and efficient evaluation by using PLMs as subject surrogates to mimic Item Response Theory, bypassing the need for human test subjects. We will provide our dataset and codes upon request.

| Related Research | Answer Type | Dataset | Factors to Control/Generate | | | Difficulty Control (Evaluation Method) | Difficulty Level |
|---|---|---|---|---|---|---|---|
| | | | Distractor (Selection Method) | Gap (Generation Method) | Stem | | |
| Gao et al. 2019a | R. C. | SQuAD | | | √ | Yes (RC system) | Item Level |
| Gao et al. 2019b | R. C. | RACE | √ | | | None | |
| Chung et al. 2020 | R. C. | RACE | √ | | | None | |
| Qiu et al. 2020 | R. C. | RACE | √ | | | None | |
| Felice et al. 2022 | Open Cloze | private | | √ (Electra) | | None | |
| Matsumori et al. 2023 | Open Cloze | private | | √ (gap score) | | None | |
| Lee et al. 2019 | C-test | Beiborn et al.2016 | | √ (prediction) | | Yes (Human Subject) | Item Level |
| Lee et al. 2020 | C-test | Beiborn et al.2016 | | √ (entropy) | | Yes (MLP model) | Proficiency Level |
| Susantia et al. 2017 | MC Cloze | TOEFL iBT | √ (feature-based) | | √ | Yes (Human subject) | Item Level |
| Yeung et al. 2019 | MC Cloze | Chinese sentences | √ (BERT-based ranking) | | | None | |
| Ren and Zhu, 2021 | MC Cloze | DGen | √ (featured-based L2R) | | | None | |
| Panda et al. 2022 | MC Cloze | ESL lounge | √ (BERT-based and feature-based) | | | None | |
| Chiang et al. 2022 | MC Cloze | CLOTH, DGen | √ (BERT-based and feature-based)) | | | None | |
| Wang et al. 2023 | MC Cloze | CLOTH, DGen | √ (Text2Text) | | | None | |
| **Our Research** | **MC Cloze** | **CLOTH** | **√ (BERT-based and feature-based with validity rules)** | **√ (confidence-based entropy)** | | **Yes (PLM-based IRT Assessment)** | **Item Level** |

Table 2: Recent Research on Question Generation for Language Proficiency Test

## 2 Related Research

The language proficiency test commonly adopts cloze tests (open or multiple-choice), C-tests, and reading comprehension (RC) to assess students' language skills. Question Generation (QG) aims to create natural and human-like questions from diverse data sources. Research on MC cloze test question generation primarily focuses on tasks such as analyzing factors influencing item difficulty (Susanti et al., 2017), distractor generation (Yeung et al., 2019; Ren and Zhu, 2021; Chiang et al., 2022), and reducing invalid distractors (Zesch and Melamud, 2014; Wojatzki et al., 2016). Table 2 presents a comparative analysis of recent studies on the automatic generation of cloze test, RC, and C-test.

For MC cloze test, distractor generation algorithms aim to identify plausible but incorrect candidates for filling in blanks. Selection is based on semantic proximity to the target word, measured through methods like WordNet (Brown et al., 2005), thesauri (Smith et al., 2010), and word embeddings similarity (Guo et al., 2016; Susanti et al., 2015; Jiang and Lee, 2017). Recent studies utilize confidence scores from BERT models (Devlin et al. 2018) for ranking distractor candidates, outperforming semantic similarity methods in correlation with human judgment (Yeung et al., 2019). Ren and Zhu (2021) apply knowledge-based techniques to help generate distractor candidates. Chiang et al. (2022) suggest BERT-based methods as superior in distractor generation. Their candidate selection relies on confidence scores from pretrained language models. Wang et al. (2023) propose a Text2Text formulation using pseudo Kullback-Leibler divergence, candidate augmentation and multi-task training, enhancing performance in generating distractors that align with benchmarks.

Item difficulty is crucial in adaptive testing, yet few studies focus on generating items with diverse difficulty levels different from standard benchmark datasets. Furthermore, these works typically rely on human test-taker evaluations (Susanti et al., 2017; Lee et al., 2019). A few studies used model judgments in RC test (Gao et al., 2019) and C-test (Lee et al., 2020). In related research on question difficulty estimation, QA models are also proposed to estimate difficulty through item response theory (Benedetto, 2022).

Gap generation has been the focus in the context of C-tests (Lee et al. 2019 and 2020). In open cloze tests, Felice et al. (2022) recommend transformer models and multi-objective learning for gap prediction. Matsumori et al. (2023) propose a masked language model approach with a gap score metric for generating open cloze questions tailored to specific target words. In contrast, research addressing the control of difficulty levels by modifying both distractors and gaps in multiple-choice cloze tests is lacking.

# 3 Methodology

Our research addresses key challenges in generating MC cloze questions. We aim to produce questions with varying difficulty by managing both the gap and distractors, using ranking rules to eliminate invalid distractors. We also propose a PLM-based IRT assessment framework to objectively evaluate item-level difficulty changes, alleviating reliance on human annotation.

As shown in Figure 1, our approach involves: (1) training PLM-based models on benchmark data to simulate test-takers; (2) designing strategies to control difficulty by manipulating gaps and distractors; (3) using PLM-based surrogate models to take the modified tests and applying IRT to evaluate difficulty changes.

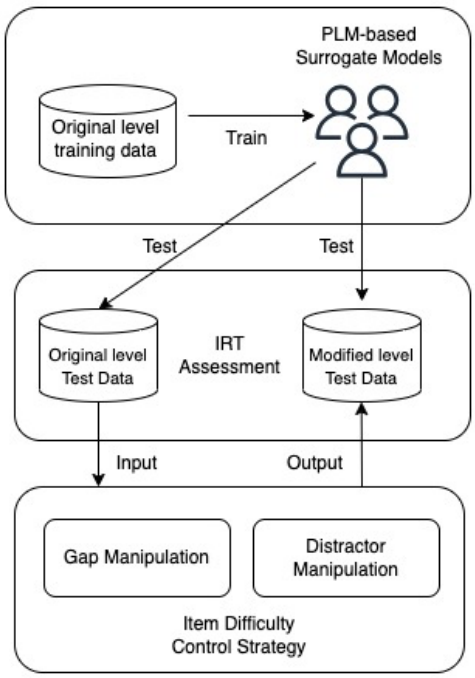


Figure 1: Research Structure

## 3.1 IRT Assessment with PLM-based Surrogate Models

Calibrating test difficulty traditionally requires trials with human subjects, which is time-consuming and costly. IRT is a framework to estimate item difficulty unsupervised (Benedetto, 2022; Susanti et al., 2017). Previous work used Reading Comprehension systems or MLP models to evaluate change of predictions (Gao et al. 2019a, Lee et al. 2020). We propose that predictions by various PLMs with different settings can simulate human test-taking for IRT without actual test-takers.

We fine-tune 12 PLM models on each dataset using BigBird (Zaheer et al. 2020) and Electra (Clark et al. 2020) with different hyperparameters. Control strategies generate hard and easy versions of each test fold. Trained surrogate models take these versions, and their scores are aggregated across folds. An IRT model fitted on the aggregated scores for the original and modified tests evaluates difficulty shifts between easy and hard versions by modeling score distributions.

## 3.2 Difficulty-controllable Question generation

For difficulty-controllable question generation, we combine PLM-based confidence scores, semantic similarity and edit distance metrics, and validity rules to generate gaps and distractors at tunable difficulty levels.

**Gap Difficulty Control**:

Entropy has been studied as a proxy for gap complexity in open cloze tests (Felice et al., 2022) and C-tests (Lee et al., 2020). We propose leveraging pre-trained model confidence scores for entropy estimation of candidate gaps, without separate training (Figure 10, Appendix B).

Given a cloze stem, we identify candidate gap words matching the POS tag of the original key. We fine-tune a model like BERT to predict words for each candidate gap. For each gap, we calculate the Shannon entropy using the top K predictions ordered by the model's confidence score:

$$H(X) = -\sum_{i=1}^{n} p(x_i)\log_2 p(x_i)$$

where $x_i$ is the $i_{th}$ word predicted by BERT for the candidate gap, and $p(x_i)$ is BERT model's confidence score for $x_i$ .

We sort candidate gaps by entropy and select high entropy ones for hard questions and low entropy for easy questions. Hard questions are generated by selecting hard gaps and generating more difficult distractors for the selected gaps, and vice versa for easy questions.

**Distractor Difficulty Factors**:

Inspired by previous work (Susantia et al. 2017, Yeung et al. 2019, Ren and Zhu 2021, Chiang et al. 2022), we design three factors for distractor generation: semantic similarity using word2vec cosine similarity, syntactic similarity using Levenshtein distance, and PLM confidence scores for gap prediction.

- **Confidence score**

Formally, let $\mathbb{M}(\,)$ be a PLM model finetuned with our training data set, $S$ be a cloze stem, $V$ be a vocabulary list, $A$ be the answer of $S$, and $d_i$ be a word in $V$ as a candidate distractor. We denote a given stem $S$ with the cloze blank filled in $[Mask]$ with $S_{\otimes[Mask]}$.

Confidence score $C_i$ for $d_i$ given by PLM is defined:

$$C_i = p(d_i|\mathbb{M}(S_{\otimes[Mask]})$$

- **Semantic similarity**

The semantic similarity $S_i$ of the candidate distractor and the answer is defined as:

$$S_i = CosineSimilarity(Embed(d_i), Embed(A))$$

where $Embed(\,)$ refers to Glove Embedding.

- **Levenshtein ratio**

The Levenshtein ratio measures string similarity on a scale from 0 to 1. It is defined as:

$$Levenshtein_{ratio} = \frac{sum - ldist}{sum}$$

where *sum* is the total length of two strings, and *ldist* is the weighted edit distance between two strings based on Levenshtein distance (Levenshtein et al. 1966). The Levenshtein distance counts insertions, deletions, and substitutions to transform one string into the other. The weighted distance is calculated as:

$$ldist = Num(INSERT) + Num(DELETE) + 2 * Num(REPLACE)$$

**Invalid Distractor Control**

Challenging distractors are semantically similar to correct answers, but selection based solely on semantic scores or PLM confidence may generate invalid distractors that plausibly fit the gap. Previous work suggests context-sensitive lexical inference rules can filter potentially appropriate distractors (Zesch and Melamud, 2014). Our analysis reveals that fine-tuning BERT and ranking distractors by confidence scores produces 302 items with 482 invalid distractors, 365 of which ranked higher than the correct answer.

Motivated by these observations and previous research (Zesch and Melamud, 2014), we design validity rules:

(1) Valid distractors have lower confidence ranking than answers. (2) For hard items, top two distractors by semantic similarity and Levenshtein ratio are selected from 50 ranks after the answer. (3) For easy items, bottom two are chosen from ranks 50-100 after the answer.

Annotation and validity rule impact analysis are in Appendix C.

**Distractor Selection Strategy**

With the three defined control factors and validity rules, we design two strategies for generating challenging or easy distractors. For distractor generation, we use BERT to rank and score all candidate words, then select the top 100 ranks after the correct answer to form the distractor candidate list, implementing the first validity rule.

The first strategy, Confidence-Ranking Control (Figure 11, Appendix B), chooses the top 3 highest-confidence distractors from the candidate list for difficult questions and the bottom 3 for easier questions.

The second strategy, 3-Factor Ranking Control, combines all three control factors – confidence scores, word2vec embedding similarity, and Levenshtein distance – along with validity rules 2 and 3 (Figure 12, Appendix B). This integrated approach allows us to tune distractor difficulty.

## 4 Experiment Design

This section presents the experimentation details for validating our proposed framework and methods. Table 6 provides generation examples referencing the original item shown in Table 1.

| | Distractor generation w/ Confidence-Ranking Control | Distractor generation w/ 3-Factor Ranking Control |
|---|---|---|
| Hard | (I)<br>**Stem:**<br>I knelt and put my arms around the child. Then the tears came, slowly at first , but soon she was ___ her heart out against my shoulder.<br>**Options:**<br>A. crying B. sobbing C. pouring D. weeping<br>**Key:** A **Distractors:** B C D | (II)<br>**Stem:**<br>I knelt and put my arms around the child. Then the tears came, slowly at first , but soon she was ___ her heart out against my shoulder.<br>**Options:**<br>A. crying B. screaming C. cried D. crushed<br>**Key:** A **Distractors:** B C D |
| Easy | (III)<br>**Stem:**<br>I knelt and put my arms around the child. Then the tears came, slowly at first , but soon she was ___ her heart out against my shoulder.<br>**Options:**<br>A. crying B. counting C. shouting D. booming<br>**Key:** A **Distractors:** B C D | (IV)<br>**Stem:**<br>I knelt and put my arms around the child. Then the tears came, slowly at first , but soon she was ___ her heart out against my shoulder.<br>**Options:**<br>A. crying B. owing C. caves D. sobbed<br>**Key:** A **Distractors:** B C D |

Table 3: Generated hard and easy items for the original item in Table

### 4.1 Dataset

Various datasets have been used for cloze test generation (Table 1), with CLOTH[1] (Xie et al., 2017) and DGen (Ren & Zhu, 2021) being popular choices. DGen compiles science questions from diverse sources and levels, while CLOTH contains cloze-style English reading comprehension questions for middle-school and high-school entrance exams. We selected CLOTH as it aligns closely with our goal of controlling item difficulty for adaptive testing. We strictly follow the "Terms and Conditions" as listed on the download site.

We divided the CLOTH dataset into two sets according to its two proficiency levels – CLOTH-M for middle school and CLOTH-H for high school entrance exams. Each set was further segmented into 5 folds. Within each fold, we split the passages into stems. Stems comprised consecutive sentences leading up to the first [MASK] token (i.e. gap), ensuring sufficient context surrounding the cloze deletion. Data statistics is provided in Appendix A.

### 4.2 Evaluation

For each data fold, we trained 12 PLM models using BigBird and Electra architectures, with learning rates of 1e-4, 1e-5, and 3e-5, batch sizes of 16 and 32, epoch of 1 and AdamW optimizer. We conducted experiments on a single NVIDIA Quadro RTX 8000 GPU. The control strategies were applied to the "Test" split. By concatenating the scores across all surrogate models and test folds, IRT models were then fitted to quantify overall changes in test difficulty. We use the py-irt library (Lalor and Rodriguez, 2023) as it leverages PyTorch and GPU acceleration for faster and more scalable IRT modeling compared to existing libraries. We apply the 1PL (also known as the Rash model) with default setting. This model estimates a latent ability parameter for subjects and a latent difficulty parameter for items, which fits exactly what we intend to evaluate.

## 5 Results and Analysis

**Surrogate Model Performance**

Table 3 presents the surrogate models' average accuracies across the five test data folds on the original cloze items. Italic numbers indicate the 12 CLOTH-M surrogates' performances, while underlined numbers show the 12 CLOTH-H surrogates. The models exhibit a wide accuracy range (0.42 to 0.81), demonstrating diverse capabilities as artificial test takers for difficulty modeling.

| Proficiency | CLOTH-M | | CLOTH-H | |
|---|---|---|---|---|
| Model | BigBird | Electra | BigBird | Electra |
| 1e-4，16 | *0.4282* | *0.7106* | 0.4234 | 0.527 |
| 1e-4，32 | *0.6691* | *0.7306* | 0.5671 | 0.6601 |
| 1e-5，16 | *0.811* | *0.7613* | 0.7902 | 0.7119 |
| 1e-5，32 | *0.8081* | *0.7602* | 0.7974 | 0.7102 |
| 3e-5，16 | *0.6093* | *0.7558* | 0.687 | 0.7008 |
| 3e-5，32 | *0.798* | *0.7615* | 0.7814 | 0.7072 |

Table 4. Surrogate models' performance

To further analyze the surrogate models, we select 4 as middle school surrogates (Electra (1e-4, 16), Electra (1e-4, 32), Bigbird (1e-4, 32), and Electra (3e-5, 32)) and 3 as high school surrogates

[1] https://www.cs.cmu.edu/~glai1/data/cloth/

(Bigbird (1e-5, 16), Bigbird (1e-5, 32), Bigbird (3e-5, 32)). These models are trained similarly and tested on both CLOTH-M and CLOTH-H. The table below compares the average accuracies, standard deviations, and utility ratios of the 12-surrogate sets and the middle and high school surrogate subsets:

| Model | CLOTH-M | | | CLOTH-H | | |
|---|---|---|---|---|---|---|
| | Avg. Acc. | Stdv | Utility Ratio | Avg. Acc. | Stdv | Utility Ratio |
| 12 | 0.717 | 0.104 | 73.9% | 0.672 | 0.108 | 72.1% |
| 4-mid | 0.718 | 0.034 | 38.2% | 0.615 | 0.072 | 52.2% |
| 3-high | 0.803 | 0.006 | 10.4% | 0.79 | 0.007 | 10.9% |

Table 5. Comparing surrogate models

The utility ratio is the percentage of test questions remaining after excluding those answered correctly or incorrectly by all test takers. The 4 middle school surrogates perform better on CLOTH-M and worse on CLOTH-H, while the 3 high school surrogates substantially outscore them on CLOTH-M. The smaller standard deviations demonstrate these sets represent distinct proficiency levels. The 12-surrogate sets achieve higher utility ratios (73.9%, 72.1%) than the middle and high school sets, and are retained for evaluating item difficulty control given their better utility and diverse performances to distinguish between stronger and weaker students.

## Performance of Control Methods

Figures 2 and 3 present the effect of generating difficult and easy items using the Confidence-ranking algorithm and 3-Factor strategy. The red lines show IRT distributions for difficult generated items, the blue lines for easy items, and the black dotted lines mark the original test difficulty.

Both strategies systematically manipulated cloze item difficulty. Across CLOTH-M and CLOTH-H, the strategies successfully generated harder items (red distribution shift right) and easier items (blue shift left) compared to the original test items.

However, CLOTH-H exhibits a narrower spread between high and low difficulty items, indicating greater efficacy adjusting difficulty for the lower proficiency CLOTH-M rather than the advanced CLOTH-H.

## Effect of Gap Control

The effect of gap position control on difficulty control differs for intermediate (CLOTH-M) versus advanced (CLOTH-H) questions. For CLOTH-M, retaining the original gap position versus modifying it does not substantially impact the efficacy of confidence ranking or 3-factor ranking (Fig. 4). The item difficulty distributions remain relatively consistent.

In contrast, for CLOTH-H, retaining the original gap positions leads to wider item difficulty distributions when generating hard questions, as shown by the rose dashed line and the red dashed line in Figure 4. However, for generating easy CLOTH-H items, controlling the gap position or not produces similar difficulty distributions.

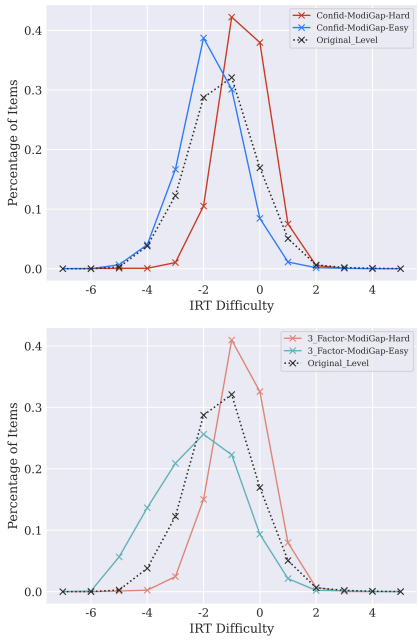


Figure 2. Change of IRT for CLOTH-M with Confidence-Ranking (above) and 3-Factor Ranking Control (below)

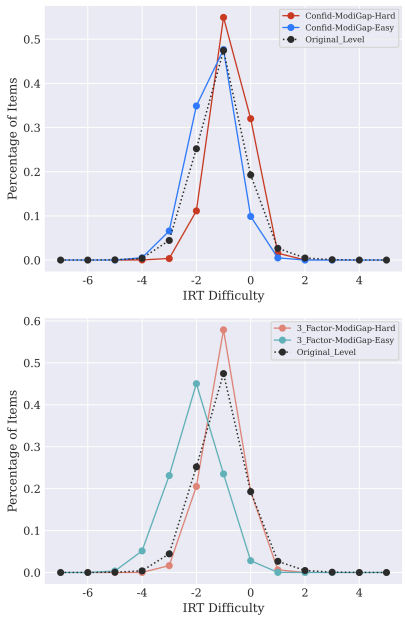


Figure 3. Change of IRT for CLOTH-H with Confidence-Ranking (above) and 3-Factor Ranking Control (below)

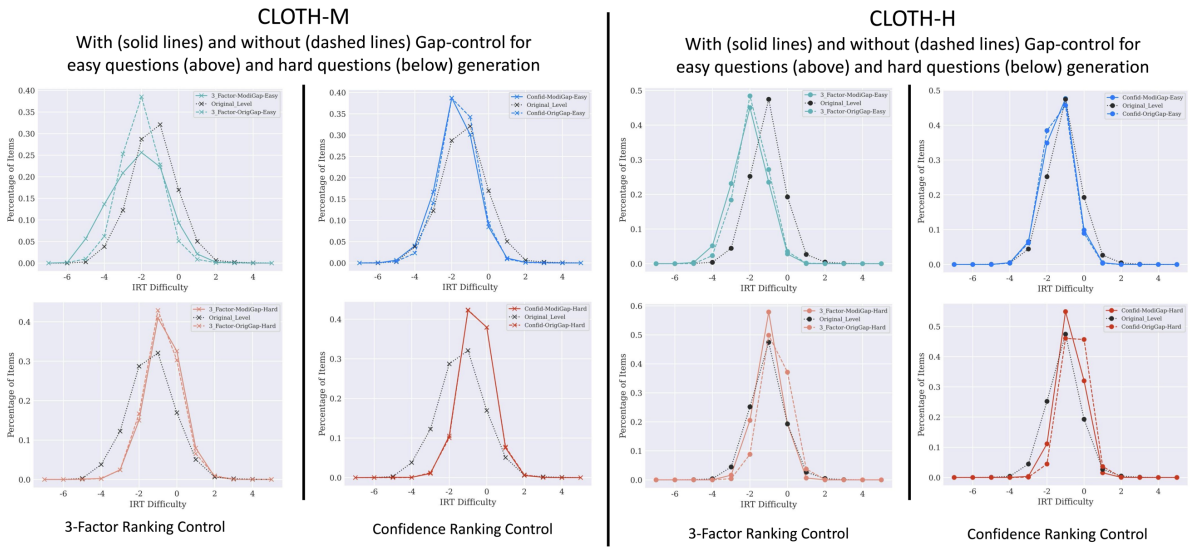


Figure 4: Comparing the Effect of Gap-control on Two Strategies for Two Datasets

## Comparing Confidence-ranking and 3-Factor Ranking on Generating Easy and Hard Items

Comparing the two control strategies, 3-factor ranking generates a slightly wider range of easy item difficulties for both datasets as shown in Figure 5. Meanwhile, confidence-ranking method without gap control produces slightly wider distributions of hard items for both datasets as evident in Figure 6.

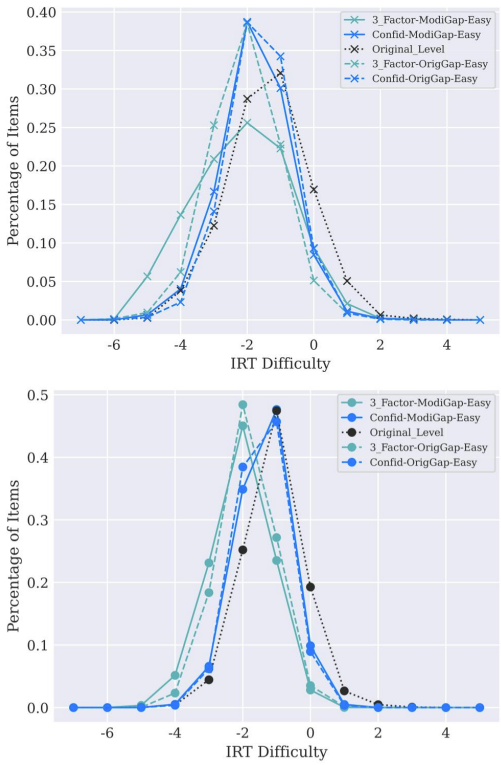


Figure 5: Confidence-ranking and 3-factor ranking w/ and w/o Gap control on generating easy items for CLOTH-M (above) and CLOTH-H (below)

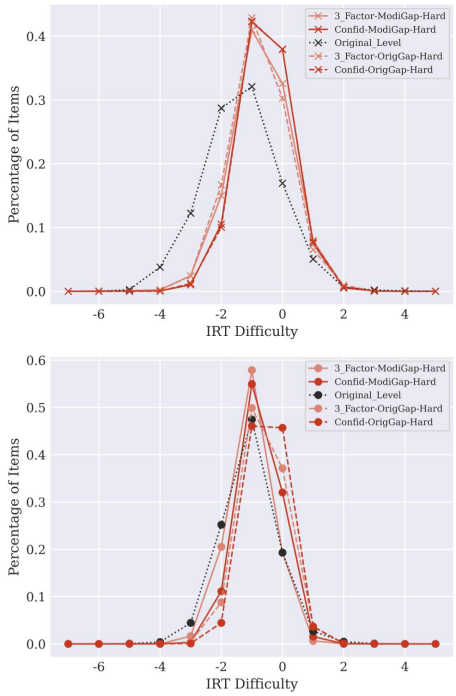


Figure 6: Confidence-Ranking and 3-Factor Ranking w/ and w/o Gap control on generating hard items for CLOTH-M (above) and CLOTH-H (below)

## Best Combination Strategies and Advantage of Gap Control

We provide the box plot analysis on best combination control strategies for the two proficiency tests. Figures 7 and 8 show that the best strategy combination is the 3-Factor Ranking control without gap for easy question and confidence ranking control without gap for hard questions. Figure 9 shows Gap Control with 3-Factor ranking enhances easy CLOTH-M item generation over 3-Factor without gap control.

While maintaining the same mean difficulty, Gap Control increases variability, indicating improved ability to span multiple difficulty values - better fulfilling key needs when creating easy test questions.

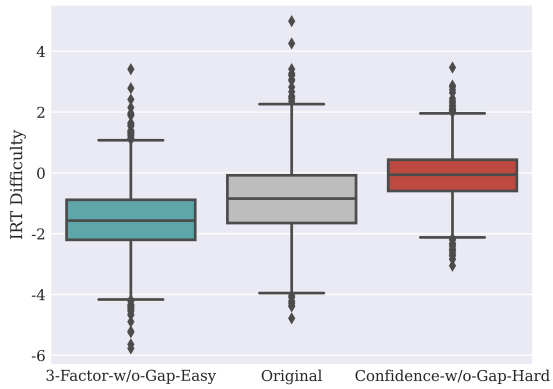


Figure 7: Best combination for CLOTH-M: 3-Factor Ranking without Gap Control for Easy Items and Confidence-Ranking without Gap Control for Hard Items Generation.

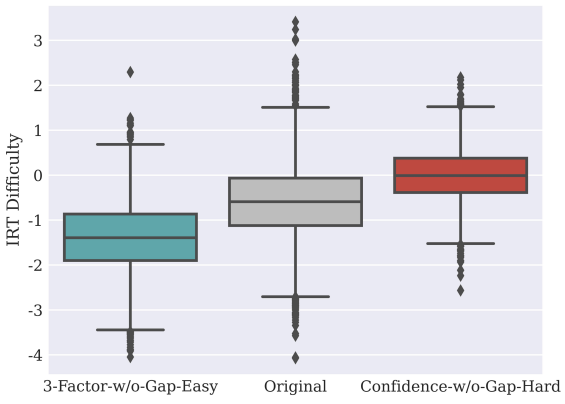


Figure 8: Best combination for CLOTH-H: 3-Factor Ranking without Gap Control for Easy Items and Confidence-Ranking without Gap Control for Hard Items Generation.

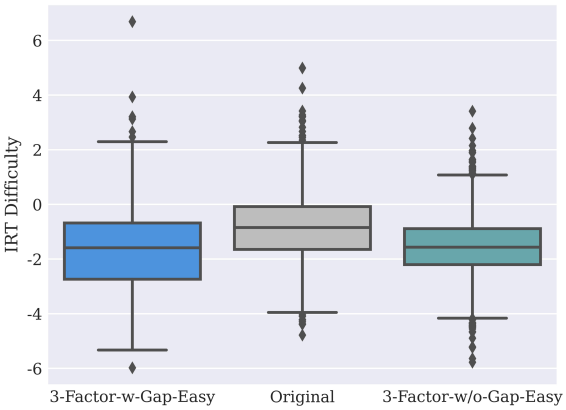


Figure 9: Gap Control strategy increases item variability than without gap control for 3-Factor Ranking method on easy item generation for CLOTH-M.

## 6 Conclusions

In this work, we proposed a novel evalutation framework for assessing control of item-level difficulty for MC Cloze test. By using diverse pretrained models as surrogate test takers, we fitted IRT distributions to quantify changes in difficulty - avoiding reliance on human test subjects.

We designed two strategies leveraging entropy, semantic similarity, edit distance to manipulate both the gap position and distractor selection for difficultuy-controlled question generation. We further implemented validity rules to reduce generation of invalid distractors.

Systematic experimentation shows: (1) The advanced test (CLOTH-H) is more difficult to control than intermediate test (CLOTH-M); (2) Gap control has a limited effect, yet increases item variability for easy CLOTH-M generation; (3) Comparatively, 3-Factor Ranking Control method works better for easy items generation while Confidence Ranking Control method exceeds at hard item generation; (4) Validity rules reduce but do not eliminate invalid distractors -- further study into this challenge is desired.

## 7 Limitation

Our difficulty control methods worked better for intermediate exam questions than advanced ones. More research is needed to improve the methods' ability to handle very complex test items. Additionally, our techniques should be validated across other subject domains. Questions also persist around optimizing validity methods to avoid invalid distractors. We do not anticipate any potential risks or ethical concerns arising from the proposed framework for generating multiple-choice cloze test questions with controllable difficulty levels using pre-trained language models.

## References

Benedetto, L. 2022. An assessment of recent techniques for question difficulty estimation from text.

Brown, J., Frishkoff, G., & Eskenazi, M. 2005. Automatic question generation for vocabulary assessment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing* pages 819-826.

Chiang, S. H., Wang, S. C., & Fan, Y. C. 2022. CDGP: Automatic Cloze Distractor Generation based on Pre-trained Language Model. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5835–5840.

Chung, H. L., Chan, Y. H., & Fan, Y. C. 2020. A BERT-based distractor generation scheme with multi-tasking and negative answer training strategies. *arXiv preprint arXiv:2010.05384*.

Clark, K., Luong, M. T., Le, Q. V., & Manning, C. D. 2020. Electra: Pretraining text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555.*

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Felice, M., & Buttery, P. 2019. Entropy as a Proxy for Gap Complexity in Open Cloze Tests. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 323–327.

Felice, M., Taslimipoor, S., & Buttery, P. 2022. Constructing open cloze tests using generation and discrimination capabilities of transformers. *arXiv preprint arXiv:2204.07237*.

Gao, Y., Bing, L., Chen, W., Lyu, M. R., & King, I. 2019(a). Difficulty controllable generation of reading comprehension questions. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 4968–4974.

Gao, Y., Bing, L., Chen, W., Lyu, M. R., & King, I. 2019(b). Generating Distractors for Reading Comprehension Questions from Real Examinations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01): pages 6423–30.

Guo, Q., Kulkarni, C., Kittur, A., Bigham, J. P., & Brunskill, E. 2016, May. Questimator: Generating knowledge assessments for arbitrary topics. In *IJCAI-16: Proceedings of the AAAI Twenty-Fifth International Joint Conference on Artificial Intelligence*.

Jiang, S., & Lee, J. S. 2017. Distractor generation for chinese fill-in-the-blank items. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications* pages 143-148.

Lalor, J. P., & Rodriguez, P. 2023. py-irt: A scalable item response theory library for python. *INFORMS Journal on Computing*, *35*(1), pages 5-13.

Lee, J. U., Schwan, E., & Meyer, C. M. 2019. Manipulating the Difficulty of C-Tests. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 360–370.

Lee, J. U., Meyer, C. M., & Gurevych, I. 2020. Empowering active learning to jointly optimize system and user demands. *arXiv preprint arXiv:2005.04470*.

Levenshtein, V. I. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady* Vol. 10, No. 8, pages 707-710.

Matsumori, S., Okuoka, K., Shibata, R., Inoue, M., Fukuchi, Y., & Imai, M. 2023. Mask and Cloze: Automatic Open Cloze Question Generation Using a Masked Language Model. *IEEE Access*, *11*, pages 9835-9850.

Panda, S., Gomez, F. P., Flor, M., & Rozovskaya, A. 2022. Automatic Generation of Distractors for Fill-in-the-Blank Exercises with Round-Trip Neural Machine Translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 391–401.

Qiu, Z., Wu, X., & Fan, W. 2020. Automatic distractor generation for multiple choice questions in standard tests. *arXiv preprint arXiv:2011.13100*.

Ren, S., & Zhu, K. Q. 2021. Knowledge-driven distractor generation for cloze-style multiple choice questions. In *Proceedings of the AAAI conference on artificial intelligence,* Vol. 35, No. 5, pages 4339-4347

Smith, S.; Avinesh, P.; and Kilgarriff, A. 2010. Gap-fill tests for language learners: Corpus-driven item generation. In *Proceedings of ICON-2010: 8th International Conference on Natural Language Processing*, pages 1–6.

Susanti, Y. Iida, R. and Tokunaga, T. 2015. Automatic Generation of English Vocabulary Tests. In *Proceedings of the 7th International Conference on Computer Supported Education*, pages 77-87.

Susanti, Y., Tokunaga, T., Nishikawa, H., & Obari, H. 2017. Controlling item difficulty for automatic vocabulary question generation. *Research and practice in technology enhanced learning*, 12(1):1–16.

Wang, H. J., Hsieh, K. Y., Yu, H. C., Tsou, J. C., Shih, Y. A., Huang, C. H., & Fan, Y. C. 2023. Distractor Generation based on Text2Text Language Models with Pseudo Kullback-Leibler Divergence Regulation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12477–12491.

Wojatzki, M., Melamud, O., & Zesch, T. 2016. Bundled Gap Filling: A New Paradigm for Unambiguous Cloze Exercises. In *Proceedings of*

*the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 172–181.

Xie, Q., Lai, G., Dai, Z., & Hovy, E. 2018. Large-scale Cloze Test Dataset Created by Teachers. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2344–2356.

Yeung, C. Y., Lee, J. S., & Tsou, B. K. 2019. Difficulty-aware Distractor Generation for Gap-Fill Items. In *Proceedings of the The 17th Annual Workshop of the Australasian Language Technology Association*, pages 159–164.

Zaheer, M., Guruganesh, G., Dubey, K. A., Ainslie, J., Alberti, C., Ontanon, S., ... & Ahmed, A. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33, pages 17283-17297.

Zesch, T., & Melamud, O. 2014. Automatic Generation of Challenging Distractors Using Context-Sensitive Inference Rules. In *Proceedings of the Ninth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 143–148.

## A Data Statistics

Table 3 presents the number of items per split in our dataset.

| Fold | Split | CLOTH-M | CLOTH-H |
|---|---|---|---|
| 0 | Train | 17123 | 42540 |
| | Validate | 5678 | 14189 |
| | Test | 5669 | 14139 |
| 1 | Train | 16975 | 42432 |
| | Validate | 5757 | 14155 |
| | Test | 5738 | 14281 |
| 2 | Train | 17011 | 42628 |
| | Validate | 5680 | 14145 |
| | Test | 5779 | 14095 |
| 3 | Train | 17094 | 42502 |
| | Validate | 5733 | 14194 |
| | Test | 5643 | 14172 |
| 4 | Train | 17077 | 42463 |
| | Validate | 5752 | 14224 |
| | Test | 5641 | 14181 |

Table 6: Data Statistics

## B Algorithms

Figures 8, 9, and 10 present the algorithms for Gap Control, Confidence-Ranking Control, and 3-Factor Ranking Control respectively.

**Algorithm 1:** Gap generation

**Input :** A sentence list with POS $A = [s_1, ..., s_n]$, Dict of POS numbers for answer words $D_{POS}$, Prediction model $M$, Numbers of model candidate $K$, Target level $L$

**Output:** Target sentence list $T$

$T \leftarrow []$;
**for** *key in* $D_{POS}$ **do**
  $AllSentenceDict[key] \leftarrow []$;
**end**
**for** *sentence in* $A$ **do**
  **for** *word in sentence* **do**
    $pos_{word} = pos(word)$;
    **if** $pos_{word}$ *in* $D_{POS}$ **then**
      $sentence_{mask} = CreateMaskSentence(sentence, word)$;
      $PredList = ModelPred(sentence_{mask}, K, M)$ ; // predict the top $K$ score of $sentence_{mask}$ Using $M$
      $shannon = CalShannon(PredList)$ ; // compute Shannon entropy of $sentence_{mask}$
      $AllSentenceDict[pos_{word}]$.append($[sentence_{mask}, shannon]$)
    **end**
  **end**
**end**
**for** *key in AllSentenceDict* **do**
  $value = D_{POS}[key]$;
  **if** $L$ *is Hard* **then**
    $T \leftarrow$ Top *value* candidate sentences sorted by *shannon* in $allSentenceDict[key]$
  **else if** $L$ *is Easy* **then**
    $T \leftarrow$ Last *value* candidate sentences sorted by *shannon* in $allSentenceDict[key]$
  **end**
**end**

Figure 10: Gap Generation Algorithm

**Algorithm 2:** Distractor Generation with Confidence Ranking

**Input :** A sentence $S$ with a cloze, Answer Word $A$, Prediction Model $M$, Vocabulary List $V$, Numbers of Candidate $K$

**Output:** Difficult distractors $D_{hard}$, Easy distractors $D_{easy}$

$D_{hard}, D_{easy} \leftarrow \{\}, \{\}$;
$ConfidenceList \leftarrow sorted(ModelPred(S, M, A, V))$ ; // Sort vocabulary $V$ based on the scores predicted by the model $M$
$Position_A \leftarrow ConfidenceList$.index$(A)$;
$CandidateList = ConfidenceList[Position_A + 1 : Position_A + K]$;
$D_{hard} \leftarrow$ Top 3 candidates by $CandidateList$;
$D_{easy} \leftarrow$ Last 3 candidates by $CandidateList$;
return $D_{hard}$, $D_{easy}$

Figure 11: Distractor Generation with Confidence-Ranking Control

**Algorithm 3:** Distractor Generation with 3-Factor Ranking

**Input :** A sentence $S$ with a cloze, Answer Word $A$, Prediction Model $M$, Vocabulary List $V$, Numbers of Candidate $K$

**Output:** Difficult distractors $D_{hard}$, Easy distractors $D_{easy}$

$D_{hard}, D_{easy} \leftarrow \{\}, \{\}$;
$ConfidenceList \leftarrow sorted(ModelPred(S, M, A, V))$ ; // Sort vocabulary $V$ based on the scores predicted by the model $M$
$Position_A \leftarrow ConfidenceList$.index$(A)$;
$CandidateList \leftarrow ConfidenceList[Position_A + 1 : Position_A + K]$;
$Similarity_{Glove}, Similarity_{Leven} \leftarrow [], []$;
**for** *word in CandidateList* **do**
  $G \leftarrow$ Calculate Glove similarity$(A, word)$;
  $L \leftarrow$ Calculate Leven similarity$(A, word)$;
  $Similarity_{Glove}$.append$(G)$;
  $Similarity_{Leven}$.append$(L)$;
**end**
$GloveSimList_{hard}, GloveSimList_{easy} \leftarrow$Similarity$_{Glove}[Position_A + 1 : Position_A + K/2], Similarity_{Glove}[Position_A + K/2 : Position_A + K]$;
$LevenSimList_{hard}, LevenSimList_{easy} \leftarrow$Similarity$_{Leven}[Position_A + 1 : Position_A + K/2], Similarity_{Leven}[Position_A + K/2 : Position_A + K]$;
$D_{hard} \leftarrow$ Top 2 candidates by $GloveSimList_{hard}$, Top 1 candidate by $LevenSimList_{hard}$;
$D_{easy} \leftarrow$ Last 2 candidates by $GloveSimList_{easy}$, Last 1 candidate by $LevenSimList_{easy}$;
return $D_{hard}$, $D_{easy}$

Figure 12: Distractor Generation with 3-Factor Ranking Control

## C Annotation for Invalid Distractor Control

We analyzed the issues of invalid distractors with human evaluation. We recruited 9 college students at the CET-6 English proficiency level as annotators. The annotators work with our research

lab on regular basis and receive subsidy for their annotation work under supervision of our administrative office.

The invalid distractors will most likely appear when generating hard items. Using BERT's confidence score ranking without validity control, we generated distractors for 4,575 items randomly selected from the CLOTH-H dataset. Manual annotation identified 1,676 items as having at least one invalid generated distractor (i.e., a distractor that could fit as an answer in the gap). As our control strategies involves ranking distractors after the answer, we identified 302 items to further test validity rules. Among the 906 distractors generated, 482 were annotated as invalid, representing an invalidity ratio of 53.2%. After applying the Confidence-Ranking Control method and 3-Factor Ranking Control method, the ratios dropped to 20.3% and 17.3% respectively (Table 7).

| Strategy | Num. of Invalid Distractors | Ratio of Invalid Distractors |
|---|---|---|
| **Confidence ranking w/o validity rules** | 482 | 53.2% |
| **Confidence-ranking Control** | 184 | 20.3% |
| **3-Factor Ranking Control** | 160 | 17.7% |

Table 7. Manual annotation of 906 distractors generated with confidence ranking w/o validity rules, and our methods of Confidence-Ranking Control and 3-Factor Ranking Control

The following are examples of items with the answer (bolded) and invalid distractors (italicized) generated by confidence ranking without validity rules. The same item with distractors generated using Confidence-ranking Control and 3-Factor Ranking Control is also shown below:

**Example #1:**

I hope I did the right thing, Mom, Alice said. I saw a cat, all bloody but alive. I [MASK] it to the vet's, and was asked to make payment immediately.

(1) Original options:
A. **carried** B. followed C. returned D. guided

(2) Distractors generated without control:
A. **carried** B. *took* C. brought D. delivered

(3) Distractors generated with 3-Factor Ranking Control:
A. **carried** B. showed C. reported D. tried

(4) Distractors generated with Confidence Ranking Control:
A. **carried** B. transported C. hauled D. rode

**Example #2:**

[MASK] this surprised him very much, he went through the paper twice, but was still not able to find more than one mistake, so he sent for the student to question him about his work after the exam.

(1) Original options:
A. **As** B. For C. So D. Though

(2) Distractors generated without control:
A. **As** B. *Because* C. Although D. Though

(3) Distractors generated with 3-Factor Ranking Control:
A. **As** B. Even C. Once D. Soon

(4) Distractors generated with Confidence Ranking Control:
A. **As** B. Realizing C. Again D. Initially

## D Instruction to Annotators for Invalid Distractor Identification.

**Instruction**: You are given a set of multiple-choice cloze test questions, each with four options. The correct answer is identified, along with three generated distractor options. Please review the choices and identify any "invalid distractors" - alternatives that contextually fit the gap as a potentially correct response, rather than an implausible one.

For example:

---

When I began planning to move to Auckland to study, my mother was worried about a lack of jobs and cultural differences. Ignoring these ____ I got there in July 2010.

A. **concerns** B. *worries*

C. fears D. considerations

---

Here, the answer is "concerns". The generated distractors include "worries". Both are grammatically correct. "Concerns" fits the semantic context only slightly better. Therefore, in this case, "worries" is considered an "invalid distractor".

Your annotation results will help assess the efficacy of our difficulty-control strategies in limiting invalid distractor generation for multiple choice cloze tests.