# Bundled Gap Filling: A New Paradigm for Unambiguous Cloze Exercises

**Michael Wojatzki**
Language Technology Lab
University of Duisburg Essen
Duisburg, Germany
michael.wojatzki@uni-due.de

**Oren Melamud**
Computer Science Department
Bar-Ilan University
Ramat-Gan, Israel
melamuo@cs.biu.ac.il

**Torsten Zesch**
Language Technology Lab
University of Duisburg Essen
Duisburg, Germany
torsten.zesch@uni-due.de

## Abstract

Cloze tests, also known as gap-fill exercises, are a popular tool for acquiring and evaluating language proficiency. A major challenge in the way of automating scoring of cloze tests is the yet unsolved problem of gap filler ambiguity. To address this challenge, we present the concept of *bundled gap filling*, along with (1) an efficient computational model for automatically generating unambiguous *gap bundle* exercises, and (2) a disambiguation measure for guiding the construction of the exercises and validating their level of ambiguity. Our evaluation shows that our proposed exercises achieve a dramatic reduction in gap filler ambiguity, while our disambiguation measure can be effectively used to discard exercises that are nevertheless too ambiguous.

## 1 Introduction

Cloze-tests (or gap-fill tasks) (Taylor, 1953) are a frequently used exercise type, where a target word in a sentence is hidden and replaced with a gap. The learner is then asked to figure out the hidden word based on its context. While gap-fill tasks are quite easy to generate, automated scoring is difficult as gaps are often significantly ambiguous (Chavez-Oller et al., 1985). For example, in the sentence *'The students have to ___ the test'* most people would accept *do*, *take*, or *pass* as valid gap-fillers. However, for reasons of practicability, in common test scenarios this ambiguity is often ignored, which may result in high error-rates even when testing native speakers (Klein-Braley and Raatz, 1982).

The ambiguity of gap-fill tasks can be countered by providing a set of answer candidates for the learner to choose from. The candidates include a single correct solution and a set of incorrect *distractors* that are –in theory– used to control the difficulty of the task. However, providing answer candidates encourages guessing and changes the nature of the task from producing a solution to recognizing a solution (Wesche and Paribakht, 1994).

Furthermore, in practice, it is far from trivial to find good distractors and to resolve the trade-off between task difficulty and task ambiguity. Distractors that have little relatedness with the sentential context of a gap are easy to reject and therefore usually yield unambiguous but very easy tasks. Alternatively, distractors that fit well into the gap are hard to tell from the correct solution, but may be in fact also valid solutions themselves, resulting in an ambiguous exercise with more than one correct answer.

In this paper, we propose a new paradigm for addressing the ambiguity problem in gap-fill exercises, which we call *bundled gap filling*. A *gap bundle* is a set of gaps in different sentences, all hiding the exact same single word. In a gap bundle exercise, the learner is presented with all of the gaps in a bundle at the same time and asked to find the single word behind all of them. Figure 1 illustrates this approach (last row) in comparison to the traditional ones (upper rows).

As generating gap bundle exercises manually would be tedious, we propose a probabilistic gap bundle disambiguation measure based on language models (Chen and Goodman, 1999). Guided by this measure, we can both automatically generate gap bundle exercises, such that ambiguity is minimized, and discard the ones that are nevertheless too ambiguous.

Proficient English speakers are expected to get (near) perfect scores in non-ambiguous gap fill exercises, but much lower scores in ambiguous exer-

| Type | Example | Ambiguity | Automatability | Nature of Task |
|---|---|---|---|---|
| gap-fill | The students have to ___ the test. | high | high | production |
| multiple-choice gap-fill | The students have to ___ the test.<br>a) **take**<br>b) fold<br>c) entertain<br>d) fry | low to moderate<br>depending on selected distractors | low<br>as it's hard to control both ambiguity and difficulty | recognition |
| bundled gap-fill | The students have to ___ the test.<br>Their cook will ___ three salmons.<br>All passengers should ___ their seats.<br>Both authors ___ credit for this. | low to moderate<br>depending on selected gaps | high<br>using the scheme proposed in this paper | production |

**Figure 1:** Comparison of three types of gap-fill exercises: (a) gap-fill, (b) gap-fill + distractors, (c) bundled gap-fill

cises. In our empirical evaluation, native and near-native English speakers achieved a mean success rate of .78 on bundled gap exercises compared to .27 on single gap exercises in an identical setting. This suggests that indeed the ambiguity of our gap bundles is dramatically lower than that of traditional single gaps.

## 2 Bundled Gap Filling Exercises

In this section, we describe (i) our proposed bundled gap filling exercises, (ii) a probabilistic model for estimating their level of ambiguity, and (iii) a scheme to automatically construct exercises, such that ambiguity is minimized.

### 2.1 Motivation

For a gap-fill exercise to be unambiguous, we wish to ensure that any word other than the given target word would be considered a highly unlikely solution by a proficient speaker under a common-sense interpretation. Unfortunately, as previously mentioned, this is commonly not the case for a single gap, e.g. *'The students have to ___ the test'* has multiple solutions including *take* and *do*. Similarly, *take* is not the only valid gap filler in the sentence *'All passengers should ___ their seats'*. However, in this case *do* would not be considered as a likely alternative, while *find* would. Therefore, when asked to find a single word that fits both gaps, a skilled speaker would probably be able to reject both *do* and *find*. In bundled gap filling exercises, we wish to take advantage of this variance in the ambiguity patterns of gaps.

### 2.2 General Approach

We make the following approximated assumptions regarding the reader of the gap-fill exercise. First, in the mind of the reader there exists some probabilistic distribution of the likely solutions of any given gap. The more ambiguous the gap, the wider this distribution is. Second, when asked to find a single word that fits two or more gaps bundled together, the reader attempts to 'compute' the joint distribution, which is the likelihood of any word to be a gap filler for all of these gaps at the same time. For a skilled reader, if the likelihood of the original target word in this joint distribution is significantly higher than any other word, then this bundled gap filling exercise in unambiguous. While impossible to predict precisely, we approximate the likelihood distribution in a speaker's mind using an $n$-gram language model, which predicts the likelihood of a gap-filler based on similar word sequences statistics observed in a large corpus.

To construct a gap bundle for a given target word, we start with a random seed sentence containing a gap hiding this word. We compute an approximation for the probabilistic distribution of its most likely gap-fillers, and then we iteratively add more sentences with gaps hiding the same target word to the bundle to make the bundle less ambiguous. To do this effectively, at each step we try to add the gap that would make the original target word stand out most in the resulting joint distribution of the gap bundle.

## 2.3 Probabilistic Modeling of Gap Ambiguity

We denote the probabilistic distribution for gap-filler words of a single gap in a sentence as:

$$P_{w \in V}(F(g) = w) = p_g(w) \tag{1}$$

$$\sum_{w \in V} p_g(w) = 1 \tag{2}$$

where $V$ is the vocabulary containing all words, $w$ is a single word, and $F(g)$ is the gap filler for gap $g$. This distribution can be estimated using a language model as we show later in Section 2.5.

Next, we make the following approximation for the joint probabilistic distribution of the word $w$ filling the gaps in a gap bundle, given that it must be the same word filling all of these gaps:

$$\begin{aligned} &P_{w \in V}(F(b) = w) \\ = &P_{w \in V}(g_1 = w, ..., g_n = w) \\ &\propto \prod_{i \in 1..n} p_{g_i}(w), \end{aligned} \tag{3}$$

where $b$ is a gap bundle that comprises the gaps $\{g_1, ..., g_n\}$.

Finally, we define our measure $D(b)$ for the disambiguation level of a gap bundle $b$, as the log of the ratio between the probability of the target word $t$ and the probability of the most likely word $w$ other than $t$:

$$D(b) = log \frac{P(F(b) = t)}{\max\limits_{w \in V \setminus \{t\}} P(F(b) = w)} \tag{4}$$

The greater this ratio, the more likely the target word compared to any other word, and accordingly we consider the gap bundle less ambiguous.[1] Based on this formalization, in the next section, we propose a scheme that can automatically construct gap bundles with high disambiguation measure values.

Figure 2 illustrates the log probability distributions of gap-filler words for two single gaps hiding the target word *take*, as well as the joint distribution of the gap bundle comprising these two gaps. The ambiguity measure is illustrated as the difference between the log probability of the word *take* and the other most probable word in the distribution. As can be seen, the combination of the two gaps in a bundle yields a much higher disambiguation level than each of its constituents.

## 2.4 Constructing Disambiguated Gap Bundles

We next describe a scheme to automatically construct disambiguated gap bundles based on our model. As input to this process we assume the following: (1) for each desired gap bundle, a given random seed sentence with a gap, $g_1$, hiding some target word $t$; (2) a given size $m$ for the desired gap bundle; and (3) a corpus $G$, where $G_t$ is the set of gaps in $G$ hiding the target word $t$.

A straightforward approach to construct a disambiguated gap bundle for $g_1$ would be to evaluate all possible gap bundles that include $g_1$ and $(m-1)$ additional gaps from $G_t$, and choose the one with the highest disambiguation measure. By restricting the bundle to gaps only from $G_t$, we make sure that $t$ is a correct answer to the exercise (i.e. a valid gap filler), and by optimizing our disambiguation measure we wish to increase the chances of a skilled speaker to identify $t$ as the *only* correct answer.

Unfortunately, the exhaustive search described above is not useful from a practical point of view as it would require $O(m^{|G_t|} \cdot |V|)$ computations. We therefore propose instead a greedy algorithm with complexity $O(m \cdot |G_t| \cdot |V|)$ that successively selects the next gap to be added to the bundle, such that its ambiguity is reduced the most:

$$g_{i+1} = \arg\max_{g \in G_t \setminus b_i}(D(b_i \cup g)), \tag{5}$$

where $b_i$ is the gap bundle after step $i$, and $g_{i+1}$ is the gap to be added in step $(i + 1)$.

Finally, a threshold on the disambiguation level of the resulting gap bundle can be used to discard bundles for which our algorithm failed to reach a satisfactory level of disambiguation.

## 2.5 Estimating Gap-filler Distributions

A critical element of our proposed approach is the ability to efficiently estimate the distribution of gap-filler words for a given gap in a sentence. Probably

[1] We use the log of the ratio for convenience and arithmetic stability.

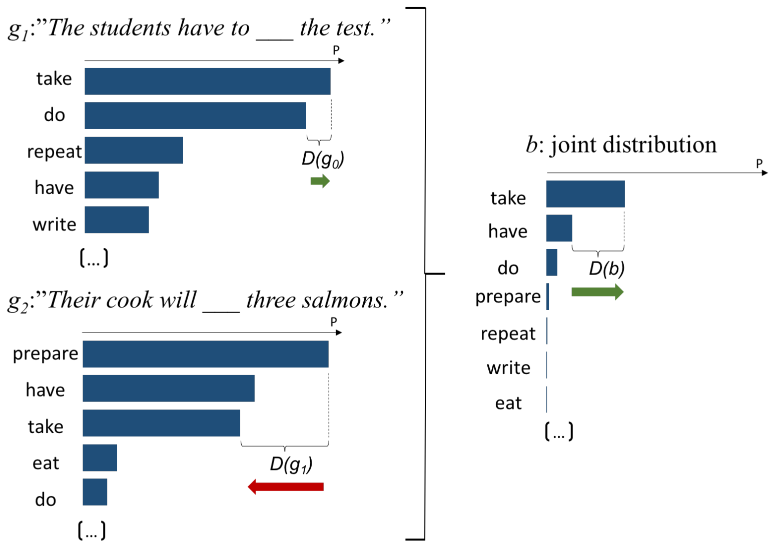

**Figure 2:** Two single gaps (top) combined into a gap bundle (bottom) for the target word *take*. The diagrams illustrate the respective gap-filler words log probability distributions. $D(g_0)$ and $D(g_1)$ are the disambiguation measures of the single gaps - note that $D(g_1)$ has a negative value. $D(b)$ is the improved disambiguation measure of the gap bundle.

the most natural choice to do this is by using probabilistic language modeling techniques, widely used in NLP for various applications (Chen and Goodman, 1999). Language models are learned from large corpora of text and used to estimate the probability of a given sequence of words, such as a sentence.

A straightforward approach to use a language model to estimate the gap-filler distribution is to fill the gap in a sentence with every possible word in the vocabulary and use the language model to estimate the probability of each resulting sentence. The probability of the gap-filler would be proportional to the probability of its respective sentence.

The problem with the above approach is that it becomes computationally intensive with large vocabularies, which may include hundreds of thousands of words. Therefore, we use instead the FASTSUBS tool (Yuret, 2012), which can efficiently compute a pruned distribution of the top-$k$ most probable gap fillers, using an $n$-gram language model. The probability of all the words below the top-$k$ is considered to be zero. For this computation to be efficient, $k$ needs to be much smaller than the size of the vocabulary. However, in practice gap fillers not in the top 1000 are usually estimated with near-zero probabilities anyway. Using pruned distributions, also brings down the complexity of our gap bundle construction algorithm from $O(m \cdot |G_t| \cdot |V|)$ to $O(m \cdot |G_t| \cdot k)$.

The main risk with pruning the distributions as described above, is that once a word is assigned with zero probability in a given gap, it will have a zero probability estimate in any bundle containing this gap, no matter how probable it is in the other gaps in the bundle. This is because the joint probability is a product of single gap probabilities. To mitigate this we use a simple form of *additive smoothing* (Chen and Goodman, 1999), that adds the probability mass of the $k$-th gap-filler to all words in the vocabulary including the ones in the top-$k$.[2]

## 3 Exercise Generation Settings

When actually generating exercises following our proposed scheme, one needs to make a few practical choices, including: (1) the corpus $C$ for training the language model, (2) the corpus $G$ from which the gaps are sampled, (3) the set of target words $T$, and (4) the number of gaps (or sentences) in a bundle.

Note that $C$ and $G$ can be the same, but $C$ would preferably be a very large corpus in order to derive a high quality language model, and $G$ would be a possibly smaller, higher quality corpus containing sen-

[2]We do not normalize the probability distributions as for our purposes we are only interested in probability ratios.

tences that are more suitable for learners.

In this section we describe some considerations related to these choice, as well as the settings used for generating the exercises in our experiments.

### 3.1 Language Model

Melamud et al. (2015) used gap-filler distributions (also known as *substitute vectors*), which are based on a language model, achieving state-of-the-art performance in lexical substitution tasks. Following their settings, we trained a 5-gram language model using the KenLM toolkit[3] (Heafield et al., 2013) with modified Kneser-Ney smoothing on the two billion word ukWaC English web corpus (Baroni et al., 2009). We then used this language model with the FASTSUBS tool[4] (Yuret, 2012) to generate the probability distribution of gap-filler words, pruned to the top 1000 most probable gap fillers.

Using a large web corpus, such as ukWaC, entails good coverage of diverse language styles for our language model, at some acceptable cost of noisy low quality texts.

### 3.2 Gap Bundle Corpus

While our algorithm can generate bundles given any set of input sentences, the quality of the exercises might vary strongly according to the corpus, from which the sentences are selected.

By selecting a corpus focused on a certain domain, a teacher may tune the generated bundles to the desired learning goals. For example, Sasaki (2000) show that participants perform better in a cloze exercise if it contains familiar cultural issues.

The corpus should be sufficiently large to offer enough distinct gaps for each target word to choose from. In contrast, if the corpus exceeds a certain size, it makes computation expensive without adding much value. In addition, the quality of exercises depends on the length of the sentences in the bundle. Short sentences provide little context for disambiguating the gap, while too long sentences are hard to parse for the learner.

As an example for a gap bundle corpus, we select the GUM corpus[5] (Zeldes, 2016) that is sufficiently large (44,000 tokens) and contains a good variety of topics. The corpus is created from 54 articles extracted from the collaboration platforms *Wikinews*, *Wikivoyage*, and *wikiHow*. The articles contain interviews, news-articles, travel guides and 'how-to' instructions, and represent a good trade-off between formal and everyday language.

### 3.3 Target Words

In principle, our algorithm works with any target word. However, as with the choice of the gap bundle corpus, the selection of target words will influence the nature of the resulting exercises.

According to Abraham and Chapelle (1992) the difficulty of a gap is influenced by whether the omitted word is a function word or a content word. In this work we choose to focus on content words.

Another major influencing factor is the frequency of the selected target words (Kobayashi, 2002). For our study, we select words from the middle of the frequency distribution avoiding the extreme ends. For very infrequent words our gap corpus will not contain enough gaps to choose from. Very frequent words are mostly function words, which may behave differently and are out of scope.

As there might still be frequency effects, we sample the target words from different frequency classes. We compute the frequency class $\hat{f}$ of a word $w$ in a given vocabulary $V$ by using:

$$\hat{f}(w) = \lfloor 0.5 - \log_2(\frac{f(w)}{f_{max}(V)}) \rfloor,$$

where $f_{max}(V)$ is the frequency count of the most frequent word in $V$. Table 1 shows the selected target words and their frequency classes.

There are additional attributes that influence the resulting gaps. For example, longer target words result in increased difficulty in traditional gap-fill tasks (Abraham and Chapelle, 1992; Brown, 1989). We leave the examination of such factors to future work.

### 3.4 Gap Bundle Size

We hypothesize that generally the more sentences with gaps we add to a bundle the less ambiguous it would be. However, larger bundle sizes may result in cognitive overload or shift the nature of the task towards a test of working memory capacity. Previous work investigating multiple-choice gap-fill exercises has shown that three distractors plus the correct

[3]http://kheafield.com/code/

[4]https://github.com/denizyuret/fastsubs-googlecode

[5]https://github.com/amir-zeldes/gum

| Target Words | Frequency Class | Word Class |
|---|---|---|
| new | 5 | Adjectives |
| best | 6 | |
| full | 7 | |
| final | 8 | |
| people | 5 | Nouns |
| language | 6 | |
| information | 7 | |
| room | 8 | |
| make | 5 | Verbs |
| want | 6 | |
| add | 7 | |
| give | 8 | |

**Table 1:** List of target words

answer is the optimal number to provide (Graesser and Wisher, 2001). Therefore, in this work we consider gap bundles that include up to four sentences.

## 4 User Study

We conducted the user study described in this section to evaluate our hypothesis that bundled gap exercises are less ambiguous than the traditional single sentence cloze items. We note that the evaluation of other educational aspects, such as learner proficiency level discrimination, were left to future work.

Gap-fill exercises are typically used with language learners, such as non-native speakers or children. However, the premise behind our study is that fully proficient speakers should be able to achieve (near) perfect scores on such exercises, provided that they are not ambiguous, i.e. that the correct answer is much more likely than any other possible solution. Accordingly, we consider high scores for proficient speakers as evidence for low levels of ambiguity in gap-fill exercises and vice versa.

### 4.1 Setup

We implement the user study using an online survey with 35 participants (20 female). To ensure a high level of language proficiency, participants either have to be native speakers of English or report a high self-assessment score (e.g. C2 CEFR level).

To measure the impact of bundle size, for each target word, we start with a single seed sentence (as in the traditional cloze test scenario) and then successively present the next sentence, chosen by our automatic algorithm for the bundle, until we reach a bundle size of four. In each step, the participant is asked to provide a single word which fits best to all of the gaps presented thus far. This setup is exemplified in Figure 3, where we show the four test steps for the target word *new*. Overall, we test this for all 12 target words (see Table 1).

| | Average Success Rate | Average Disambiguation Measure |
|---|---|---|
| Single Gap | .27 | -0.50 |
| $\text{Bundle}_2$ | .59 | 4.00 |
| $\text{Bundle}_3$ | .68 | 7.75 |
| $\text{Bundle}_4$ | .78 | 11.06 |

**Table 2:** Comparison of average success rate and our estimated disambiguation measure $D(b)$ for different bundle sizes.

For each test step, we measure average *success rate* as the ratio of participants that provided the correct target word, out of all participants. As we conducted this study with highly proficient speakers, we assume that participants who fail to provide the correct answer, have a competing answer in mind, which is also valid.

### 4.2 Results

We now report and discuss the results of the user study. We address the two major points: (1) how well gap-fill bundles resolve the ambiguity of cloze tasks, and (2) whether we can automatically discard low-quality gap bundles for quality assurance.

**Exercise Ambiguity** Our main finding is that the average success rate for single sentence cloze items is .27, while it steadily rises with every added sentence reaching .78 for a four sentence bundle - see Table 2. We also find that our average disambiguation measure grows with the size of the bundle and the success rate. These results suggests that our approach is extremely effective in reducing the ambiguity of cloze items. Still, even with four-sentence bundles we observe a non-negligible (.22) error rate. Part of this could be attributed to human performance errors, but it could also be the case that some of the bundles are still somewhat ambiguous even with four sentences.

To get more insights into this phenomenon, we conducted a detailed analysis of success rate per

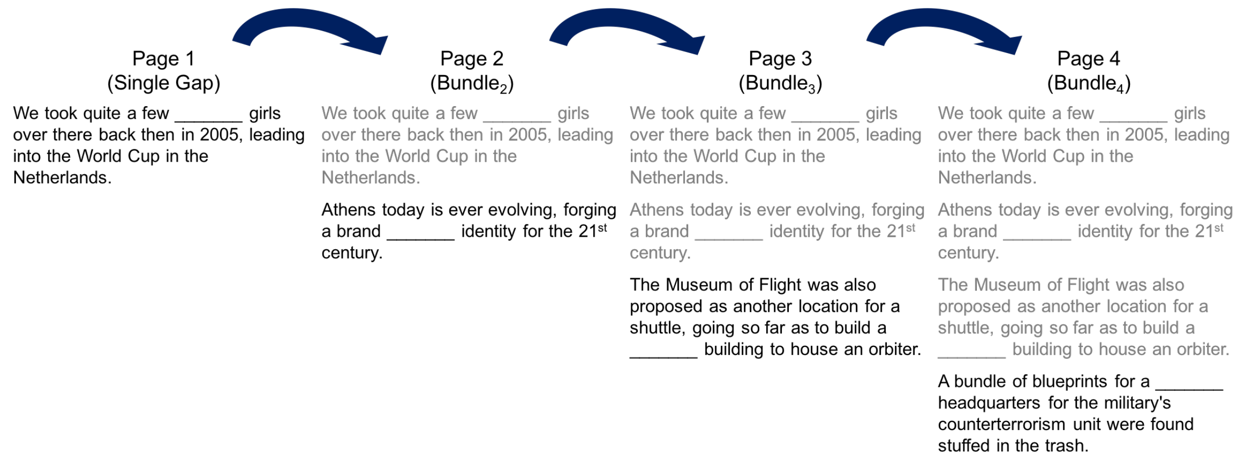
Page 1
(Single Gap)
We took quite a few _______ girls over there back then in 2005, leading into the World Cup in the Netherlands.
Page 2
(Bundle$_2$)
We took quite a few _______ girls over there back then in 2005, leading into the World Cup in the Netherlands.
Athens today is ever evolving, forging a brand _______ identity for the 21st century.
Page 3
(Bundle$_3$)
We took quite a few _______ girls over there back then in 2005, leading into the World Cup in the Netherlands.
Athens today is ever evolving, forging a brand _______ identity for the 21st century.
The Museum of Flight was also proposed as another location for a shuttle, going so far as to build a _______ building to house an orbiter.
Page 4
(Bundle$_4$)
We took quite a few _______ girls over there back then in 2005, leading into the World Cup in the Netherlands.
Athens today is ever evolving, forging a brand _______ identity for the 21st century.
The Museum of Flight was also proposed as another location for a shuttle, going so far as to build a _______ building to house an orbiter.
A bundle of blueprints for a _______ headquarters for the military's counterterrorism unit were found stuffed in the trash.

**Figure 3:** Example of the setup of the user study for the target word *new*. In the first step we present a regular single sentence cloze item, which was randomly selected. Then, we successively present larger bundles to reduce ambiguity. The subsequent sentences are selected by our algorithm.

item and bundle size - see Figure 4. On average, the biggest impact on success rate of about 30 points is observed between the single gap and the first bundle with two sentences. For some targets (*best*, *people*, *add*, and *new*) this improvement exceeds even 50 points. On the other hand, there are a few exceptions to the monotonous growth in success rate for some targets (e.g. *give*, *best* and *final*), where we observe a decline for an increased bundle size. Nevertheless, this decline is only local and does not exceed 11 points. The target *give* is particularly extraordinary as it has very high success rate across all bundle sizes. A deeper analysis shows that its randomly selected seed sentence uses the target in a quite idiomatic way ("*whales come to these protected waters to give birth.*"), which makes the gap unambiguous right from the start.

A further notable exception is the target word *final*, for which our algorithm does not manage to significantly dissolve the ambiguity (the overall improvement is just 12 points). An analysis of the variety of answers given by the study participants shows that they tended to include more adjectives that are highly related to *final* (e.g. *last* or *first*) when presented with the larger bundles. Here, the algorithm fails to provide a sentence that removes this ambiguity. We manually checked all sentences containing the word *final* in the gap bundle corpus and found that none of them could be used to really rule out *last* or *first*. This could indicate that our gap sentence corpus is not diverse enough, as it doesn't include a sentence, such as: "*This is the last and final call for flight 123 to San Diego*". However, we must also acknowledge that some words may have very close synonyms (as in *final* and *last*) that could be very hard to distinguish. In such cases, where our algorithm generates an ambiguous exercise, we wish to be able to automatically detect and discard it for quality assurance, as discussed next.

**Quality Assurance** In this analysis, we try to answer the question whether it is possible to automatically reject gap bundles that are likely to yield a low success rate due to ambiguity. We propose to use a threshold on our ambiguity measure $D(b)$ and analyze our study results to estimate the threshold value.

In Figure 5, we visualize the relationship between success rate in a given exercise and our model's corresponding disambiguation estimate $D(b)$. The estimated correlation between these two variables is decent ($r = 0.66$) though not perfect. However, a nice property with respect to thresholding is that our measure is rather conservative and does not overestimate the ambiguity reduction. This means that, for example, by discarding all items with $D(b) < 4.0$ we can remove almost all items with success rate below 50%, while keeping more than 80% of the items with success rate above 50%. This suggests that we are able to guarantee high-quality gap bundles, as long as we are allowed to reject some items (e.g. by flagging the teacher that for a certain seed sentence

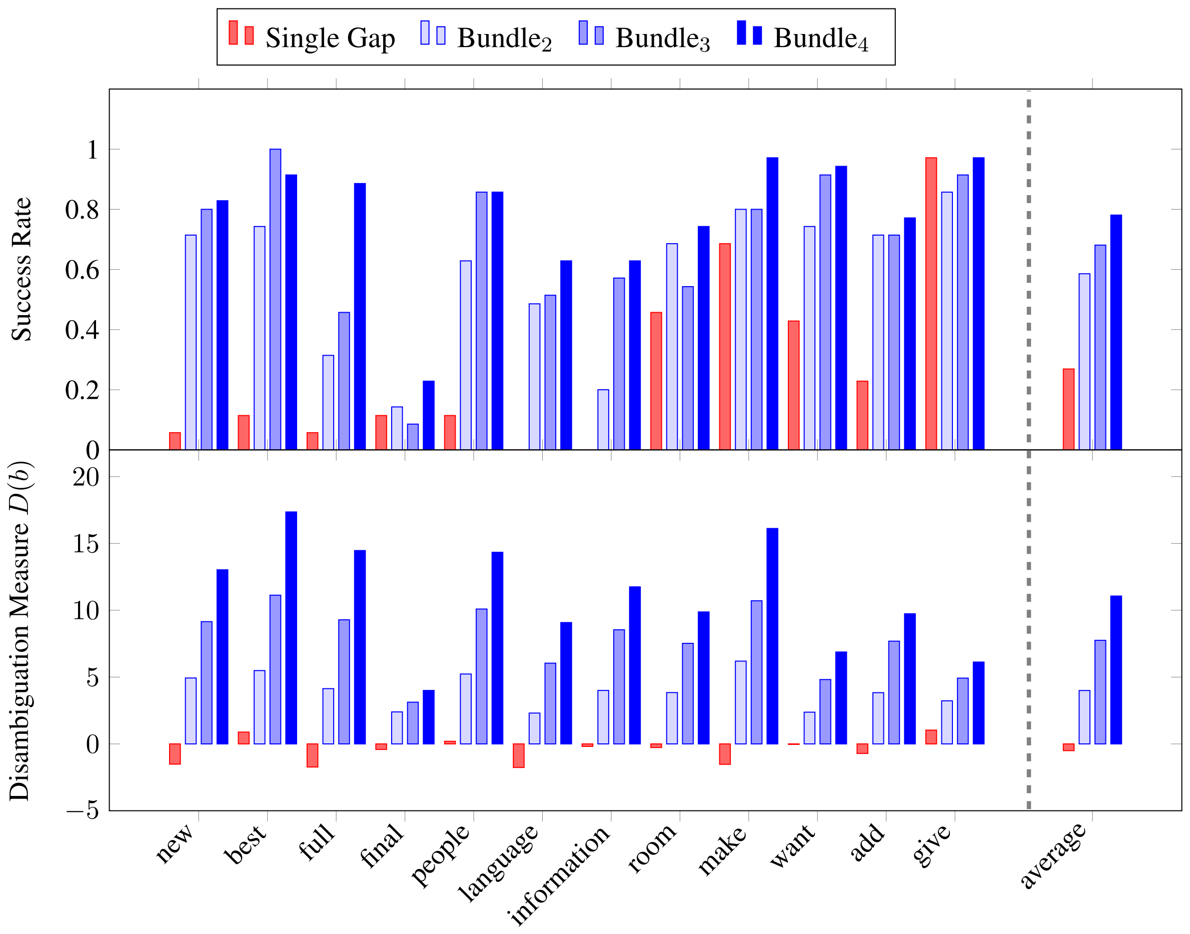

**Figure 4:** Success rate and disambiguation measure per item.

no good gap bundle can be generated).

## 5 Related Work

To the best of our knowledge, bundled gap filling is introduced in this work for the first time. Therefore, there is no directly related previous work.

However, in a broader sense, our work continues research on automated handling of ambiguity in cloze tests. Horsmann and Zesch (2014) control for ambiguity of cloze tests by selecting low ambiguity sentences based on a series of (dis-)ambiguity indicators. However, they fail to reduce the ambiguity to a sufficient degree and limit the practical relevance, as a user is limited to a narrow set of gaps that fit their approach.

As mentioned before, probably the most popular method for resolving ambiguity is to use a multiple-choice format, providing a set of distractors that may be generated automatically. These distractors may be generated from common confusions (Lee and Seneff, 2007), typical learner errors (Sakaguchi et al., 2013) or by selecting words with the same word-class or frequency in a reference corpus (Hoshino and Nakagawa, 2007).

A major problem of gap-fill exercises with distractors is that they are often too easy – especially for advanced learners –, as the generated distractors do not make sense in the context of the gap. Therefore, some approaches go one step further considering distractor candidates that are highly compatible with the context of the gaps. Then they need to automatically judge that such distractors are not in fact correct answers themselves, e.g. by considering collocations of targets and distractors (Pino and Eskenazi, 2009; Smith et al., 2010; Sumita et al., ). For example, Sumita et al. () check whether replacing the target with the distractor results in a sentence that exists on the web. If so, they conclude that the

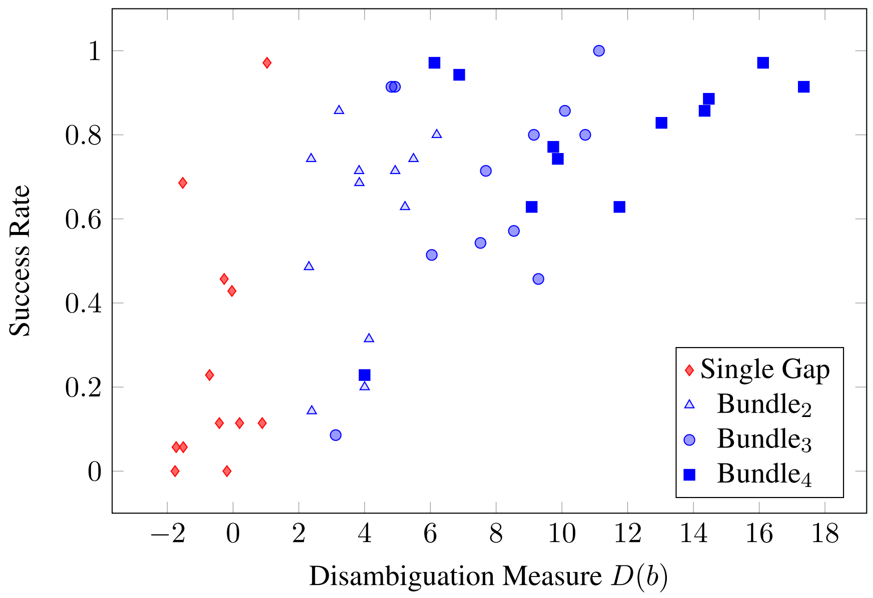

**Figure 5:** Relationship between our disambiguation measure and success rate. Each point represents a single item in the user study.

distractor is invalid. However, their approach seems to be limited, as it relies on finding exact matches of sentences which even on the web is rather unlikely. Zesch and Melamud (2014) generate distractors that are semantically similar to the target word in some sense, but not in the particular sense induced by the gap-fill context. While their approach points in a promising direction it fails to model a sufficiently large difficulty continuum, especially when targeting a group with high level of language proficiency.

Ultimately, a disadvantage of gap-fill tests with distractors is that the test is a recognition task rather than a production task and therefore considerably easier. In contrast, our *bundled gap-filling* approach has the advantage that there is no recognition stimuli and therefore the test remains a production task.

## 6 Conclusions & Future Work

In this work, we presented *bundled gap-filling* exercises and an efficient algorithm for automatically generating them. Our evaluation provides evidence that gap bundles are significantly less ambiguous than regular gap-fill exercises. This gives our approach the important advantage of supporting high automation for both generation and scoring of the exercises.

We see two main directions of research for future work. First, we want to improve the quality of the generated exercises by optimizing different parameters of the model. For example, we expect that using a larger gap sentence base or using a better language model, such as a recurrent neural network (RNN) language model, could improve the results. Second, although our bundled gap filling test is basically still a cloze test, the format change might alter the nature of the required knowledge. Consequently, we want to determine what kind of knowledge the gap bundles are actually measuring by applying it to real-life testing scenarios and examining the relations to other established measures of language proficiency. Thereby, we also want to examine the test's ability to discriminate learners with different proficiency levels by considering relations to other established measures of language proficiency and variations of the cloze test paradigm.

## Acknowledgments

This work was partially supported by the Deutsche Forschungsgemeinschaft (DFG) under grant No. GRK 2167, Research Training Group "User-Centred Social Media", by the Israel Science Foundation grant 880/12, and by the German Research Foundation through the German-Israeli Project Cooperation (DIP, grant DA 1600/1-1).

## References

Roberta G. Abraham and Carol A. Chapelle. 1992. The meaning of cloze test scores: An item difficulty perspective. *The Modern Language Journal*, 76(4):468–479.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226.

James Dean Brown. 1989. Cloze item difficulty. *JALT journal*, 11:46–67.

Mary Anne Chavez-Oller, Tetsuro Chihara, Kelley A. Weaver, and John W. Oller. 1985. When are cloze items sensitive to constraints across sentences? *Language Learning*, 35(2):181–206.

Stanley F. Chen and Joshua Goodman. 1999. An Empirical Study of Smoothing Techniques for Language Modeling. *Computer Speech & Language*, 13(4):359–393.

Arthur C. Graesser and Robert A. Wisher. 2001. Question generation as a learning multiplier in distributed learning environments. Technical report, DTIC Document.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 690–696, Sofia, Bulgaria.

Tobias Horsmann and Torsten Zesch. 2014. Towards Automatic Scoring of Cloze Items by Selecting Low-Ambiguity Contexts. In *NEALT Proceedings Series Vol.22*, pages 33–42.

Ayako Hoshino and Hiroshi Nakagawa. 2007. Assisting cloze test making with a web application. In *Proceedings of the Society for Information Technology and Teacher Education International Conference*, pages 2807–2814.

Christine Klein-Braley and Ulrich Raatz. 1982. Der C-Test: ein neuer Ansatz zur Messung allgemeiner Sprachbeherrschung. *AKS-Rundbrief*, 4:23–37.

Miyoko Kobayashi. 2002. Cloze tests revisited: Exploring item characteristics with special attention to scoring methods. *The Modern Language Journal*, 86(4):571–586.

John Lee and Stephanie Seneff. 2007. Automatic generation of cloze items for prepositions. In *Proceedings of INTERSPEECH*, pages 2173–2176, Antwerp, Belgium.

Oren Melamud, Ido Dagan, and Jacob Goldberger. 2015. Modeling Word Meaning in Context with Substitute Vectors. In *Proceedings of the NAACL*, Denver, Colorado, USA.

Juan Pino and Maxine Eskenazi. 2009. Semi-Automatic Generation of Cloze Question Distractors Effect of Students' L1. In *SLaTE Workshop on Speech and Language Technology in Education*.

Keisuke Sakaguchi, Yuki Arase, and Mamoru Komachi. 2013. Discriminative Approach to Fill-in-the-Blank Quiz Generation for Language Learners. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 238–242, Sofia, Bulgaria.

Miyuki Sasaki. 2000. Effects of cultural schemata on students' test-taking processes for cloze tests: a multiple data source approach. *Language Testing*, 17(1):85–114.

Simon Smith, PVS Avinesh, and Adam Kilgarriff. 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, Kharagpur, India.

Eiichiro Sumita, Fumiaki Sugaya, and Seiichi Yamamoto. In *Proceedings of the second workshop on Building Educational Applications Using NLP*, pages 61–68, Stroudsburg, PA, USA.

Wilson L. Taylor. 1953. "Cloze Procedure": A New Tool For Measuring Readability. *Journalism Quarterly*, 30(4):415–433.

Marjorie Wesche and Sima T. Paribakht. 1994. Enhancing Vocabulary Acquisition through Reading: A Hierarchy of Text-Related Exercise Types. Paper presented at the AAAL '94 Conference.

Deniz Yuret. 2012. Fastsubs: An efficient and exact procedure for finding the most likely lexical substitutes based on an n-gram language model. *Signal Processing Letters, IEEE*, 19(11):725–728.

Amir Zeldes. 2016. The GUM Corpus: Creating Multilayer Resources in the Classroom. *Language Resources and Evaluation*, pages 1–32.

Torsten Zesch and Oren Melamud. 2014. Automatic Generation of Challenging Distractors Using Context-Sensitive Inference Rules. In *Proceedings of the 9th Workshop on Innovative Use of NLP for Building Educational Applications at ACL*, pages 143–148, Baltimore, USA.