Automatic Cloze Generation for English Proficiency Testing

Simon Smith†, Adam Kilgarriff‡, Gong Wen-liang†, Scott Sommers‡, Wu Guang-zhong†
†National Chengchi University, Taiwan
‡Lexical Computing Ltd, UK
§Ming Chuan University, Taiwan
adam@lexmasterclass.com

Abstract
Cloze exercises are widely used in language teaching, both as a learning resource and an assessment tool. Cloze has a particular role to play in proficiency testing, where students are expected to demonstrate wide vocabulary knowledge. Cloze allows students to show that they understand the vocabulary in context, discouraging the memorization of synonyms or translations. However, it is time-consuming and difficult for item writers to make up large numbers of cloze exercises.

We present a system which automatically generates cloze exercises from a corpus. It takes the word which will form the correct answer to the exercise (the key) as input. It extracts distractors with similarities to the key from a distributional thesaurus. It then identifies a collocate of the key that does not co-occur with the distractors. Next it finds a short, simple sentence in the corpus which contains the key and the collocate. It then presents the whole item (sentence with blanked-out key, key, three distractors) to a human item-writer for approval, modification or rejection.

The system has been implemented as an application using the web API to the Sketch Engine, a leading corpus query system. We use a very large corpus (UKWaC, with 1.5 billion words) as this gives a fair-sized set of sentences to choose from for most key+collocate combinations, and allows us to infer with some confidence that, where a distractor has zero occurrences with a collocate, the combination is infelicitous. We present an initial evaluation.

Key Words: cloze, Sketch Engine, corpus linguistics, ELT, GDEX, proficiency testing

Introduction

Cloze is defined by Jonz (1990) as “the practice of measuring language proficiency or language comprehension by requiring examinees to restore words that have been removed from otherwise normal text.” The idea is traditionally attributed to Taylor (1953), when it was used as a test of text readability. The term itself derives from the concept of closure in Gestalt Theory used to describe the human tendency to mentally complete figures even when parts of that figure are missing. Taylor and other cloze researchers have used the term to describe a sample of naturally occurring text in which words are deleted and respondents asked to use semantic clues in filling in these deleted words. By the 1970s, the concept had been incorporated into educational assessment and subsequently into the assessment of English proficiency among second and foreign language learners (Alderson 1978, Oller 1973). Cloze exercises are of course widely used in proficiency testing today.

Manual Cloze Generation

It is difficult for test item writers to think up cloze exercises from scratch. They must compose or locate a convincing carrier sentence, which incorporates the desired key (the correct answer, or deleted item). They must generate distractors (wrong answers suggested to the student). This is not a trivial task, as two important constraints apply. On the one hand, the distractors must be incorrect: inserting them in the blank must generate an ill-formed sentence. On the other hand, the distractors must in some sense be viable alternatives for
completion of the carrier sentence, such as near synonyms of the key or words typically found in similar collocational contexts. An item writer encounters the following paradox: if the distractor is too distant from the key, the exercise is likely to be too easy, but if the distance is small, sentences incorporating a distractor may turn out to be infelicitously correct.

Ideally, students, teachers and testers would have an unlimited supply of cloze exercises to work with. As it takes time and expertise for a person to prepare them, we follow the lead of Mostow et al (2004) in proposing an automatic method. Our method also goes some way towards resolving the “incorrect yet viable” paradox mentioned above.

**Language Teaching meets Corpus Linguistics and Language Technology**

The task is at the intersection of language teaching, corpus linguistics and language technology (also known as computational linguistics or NLP (Natural Language Processing)). The data from which we generate the exercises is a language corpus, and the characteristics of the corpus we choose will be central to the success of the exercise, hence corpus linguistics. The techniques we use to analyze the corpus and extract what we need from it will also be critical, as will be methods for handling large corpora efficiently, hence language technology.

**The Sketch Engine**

Our system, TEDDCLOG (Taiwan English Data Driven CLOze Generation), is built on the Sketch Engine (Kilgarriff et al 2004; http://www.sketchengine.co.uk), a leading corpus query system which, in addition to concordancing and other standard features, provides word sketches—one-page summaries of a word’s grammatical and collocational behaviour—and a distributional thesaurus, which states, for each word, the words most similar to it in terms of collocation. Both these functions are critical to TEDDCLOG.

The Sketch Engine provides web access to a number of corpora (for English, Chinese and other major world languages). TEDDCLOG accesses the Sketch Engine via its web API.

**The Algorithm**

TEDDCLOG uses the following algorithm:

1. User inputs the word that will be the ‘correct answer’ for the exercise (the key)
2. Look the key up in the thesaurus to find distractors
3. Find collocates for the key in its word sketch
   a. Find a collocate which is used with the key but not with any distractors (the KOC, key-only collocate)
   b. If none can be found, return to step 2 and find another distractor
4. Find a short simple sentence containing key+KOC
5. Prepare output: blank out key from sentence, present key and distractors in random order.

Each step is described in detail below.

**The Corpus**

We use the UKWaC corpus (Ferraresi et al 2008), a very large (1.5 billion words) web corpus. Ferraresi et al, and Sharoff (2006), have shown that web corpora can give a good, broad overview of the language. The material was collected by taking a list of mid-frequency English words and sending triples of them to Google; then collecting the pages that the Google hits page listed, and using those pages as starting points for crawling. The harvested web material was then ‘cleaned’ to remove navigational material, banner advertisements, copyright statements and other noise, and duplicates and near-duplicates were removed.
These cleaning processes are important because the quality of an application such as TEDDCLOG depends on the quality and size of the corpus, and it is unlikely we will gather a corpus which is large and varied enough from anywhere other than the web. Size is important for two reasons:

1. A corpus has to be very large to provide more than a handful of sentences for most key-collocate pairings. A larger number of sentences to choose from is more likely to include short, simple ones that are suitable for language testing.
2. It is critical that the distractors are not acceptable alternatives to the key, in the context provided by the sentence. If the corpus is big enough, then the absence of any occurrences of the KOC with the distractors is strong evidence that they are not acceptable.

We have also been experimenting with the 100m word British National Corpus (BNC: http://www.natcorp.ox.ac.uk). The small evaluation described in this paper used the BNC. It would be possible to use a far larger corpus than UKWaC, by using the web as indexed by Google or Yahoo directly. This could give stronger evidence of the non-acceptability of distractors with the KOC. (Sumita et al (2005) use a method of this kind, see below.) However the use of the web in this way raises a number of other difficulties, including non-replicability of results, the limitations of the query syntax, sorting of search results and the fact that the search engines’ techniques and algorithms are unpublished trade secrets (Kilgarriff 2007), so we choose not to.

Evaluation

TEDDCLOG has been evaluated by assessing, for each of eighty input words, whether the system’s output was a viable cloze exercise. In seven cases, the judge (the first author, an experienced language teacher and native English speaker) judged that it was. The envisaged use of the system is as a cloze-exercise drafter, so the system proposes sentences and distractors which an item writer or language teacher then accepts, modifies or rejects. In this context, and in view of the prototype nature of the system, the use of BNC not UKWaC, and the cost of the traditional, manual method of cloze exercise generation, achieving seven out of 80 which are acceptable seems promising.

Related work

Several researchers have explored the possibility of automatically generating cloze exercises. Mostow et al (2004) generated cloze items of varying difficulty from children’s stories. The items were presented to children via a voice interface, and the response data was used to assess comprehension. Hoshino and Nakagawa (2007) devised an NLP-based teacher’s assistant, which first asks the user to supply a text. The system then suggests deletions that could be made, and helps the teacher to select appropriate distractors. Both of these systems use longer texts, while Sumita et al (2005) describe the automatic generation of single sentence cloze exercises from the World Wide Web. Sumita et al obtain distractors from a thesaurus, and check to make sure that there are zero Google hits for hypothesized sentences in which the key is replaced by distractors.

TEDDCLOG is similar to Sumita et al’s system in that it selects single sentences of authentic language, and looks for words with similar lexical distribution to the key to serve as distractors. However, it does not constrain its choice of distractors to synonyms, or even near synonyms; indeed, key and distractor could perfectly well be antonyms, as long as they can occur in the same contexts. Another difference between the two systems is that the Japanese team use a published resource to find distractors. We use distributional information from the
The generation algorithm of Liu, Wang & Gao (2005) starts, like TEDDCLOG, from the key. It then locates a carrier sentence including the key from a corpus. It then finds a number of distractors which, in the corpus at large, seldom collocate with important words in the target sentence. Our system differs from Liu et al.’s in that we find distractors which never occur with the key-only collocate, throughout the corpus. However, our distractors have many other collocates in common with the key.

Mostow et al. and Hoshino and Nakagawa’s systems start from texts or sentences, whereas we and Liu et al. start from the key. This is significant for two reasons: first, because item writers generally wish to use a specific word as a point of departure for producing a cloze item. Secondly, our architecture is capable of generating large numbers of cloze items on a given topic (“Business”, perhaps, or “Starting out at University”). In Smith, Sommers & Kilgarriff (2008) we reported how to extract corpora, on such topics, from the world-wide web, using WebBootCat (WBC; Baroni et al 2006). The corpora were then used to generate wordlists containing vocabulary salient to the topic. Such wordlists can readily be used as lists of keys to bootstrap collections of on-topic cloze items.

The System

First, the item writer specifies the key, or a list of keys to be processed. The writer enters the key, here the verb react, into the system. The system then finds words which have a similar lexical distribution to react, such as respond, interact, behave, realize and so on. It does this by establishing that these distractors and the key are all found with some set of other words, in specific grammatical relations: here, angrily and appropriately as modifiers, and government and people as subjects.

Next, the system finds a word which co-occurs with the key but never with the distractors: the key-only-collocate or KOC. In this example it is metal as subject: metal occurs as subject of react but not of any of the other verbs. A sentence with metal in the relation SUBJECT to react is selected from the corpus. All that remains is to delete the key from the sentence, and supply key, distractors and sentence to the test-taker in an appropriate format, as shown in Figure 1.

Thus, the carrier sentence, the key and the three incorrect answers (distractors) are returned by the system. In interactive mode, the user would be asked if they were satisfied with the item or whether they wanted to modify or reject it.

In this example, the reader will agree that only the (key) answer react is possible, and that if any of the three distractors were inserted, the word choice would be incorrect and the sentence anomalous.

We now describe each step of the algorithm in detail.

Finding distractors: the Thesaurus Module

The thesaurus module of SkE outputs words which typically occur in the same context as the search term. We show in Figure 2 the SkE Thesaurus output for react. (The figure reveals that most of the words with similar distribution to react relate to the human-interaction uses of the word, probably because this is its most frequent kind of use.) The three top-ranking list members – respond, interact and behave are noted and retained for use as distractors.
**Figure 1:** TEDCLOG system architecture

---

**Home Concordance Word List Word Sketch Thesaurus Sketch-Diff**  **Save**

**react**  ukWaC freq = 24778

<table>
<thead>
<tr>
<th>Lemma</th>
<th>Score</th>
<th>Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>respond</td>
<td>0.417</td>
<td>114163</td>
</tr>
<tr>
<td>interact</td>
<td>0.305</td>
<td>25685</td>
</tr>
<tr>
<td>behave</td>
<td>0.296</td>
<td>24508</td>
</tr>
<tr>
<td>realise</td>
<td>0.25</td>
<td>110985</td>
</tr>
<tr>
<td>cope</td>
<td>0.247</td>
<td>48313</td>
</tr>
<tr>
<td>adapt</td>
<td>0.245</td>
<td>50930</td>
</tr>
<tr>
<td>listen</td>
<td>0.238</td>
<td>127002</td>
</tr>
<tr>
<td>answer</td>
<td>0.237</td>
<td>105714</td>
</tr>
<tr>
<td>intervene</td>
<td>0.237</td>
<td>14898</td>
</tr>
<tr>
<td>contribute</td>
<td>0.235</td>
<td>137428</td>
</tr>
</tbody>
</table>

**Figure 2:** SkE Thesaurus entry for *react* (truncated)
Finding the Key-only Collocate (KOC): Sketch Differences Module

The Sketch Engine also provides a “Sketch Differences” display, showing which collocates are shared (and to what extent) and which are not, between two similar words. Figure 3 shows sketch differences for react and respond. We see that react occurs 232 times, and respond, 1624 times, with positively as a MODIFIER. The user can click on the number to see the concordance lines. (The other numbers are statistical measures of salience for the collocations, so 57.6 is the salience for react positively, 86.1 for respond positively.)

TedDCLog needs collocates (KOCs) that the key and distractors do not share. Candidate KOCs can be seen under the “react only” patterns in Figure 3. TedDCLog takes the high-salience collocates that do not occur with the first distractor.

In the simplest case, the first candidate KOC does not co-occur with any of the three distractors. In other cases, TedDCLog tests further candidate KOCs until one is found that does not occur with any of the distractors. If no such KOC can be found, the word with the next-highest salience is taken from the Thesaurus list, and a KOC with respect to that distractor is sought. The process is continued until we have a KOC, and set of distractors that do not occur with it.

At this point in the algorithm, we have decided on the key and three distractors. We have also established that we wish our carrier sentence to include a particular collocation: in our example, metals react. The next step is to determine what the carrier sentence will be.

Selection of Carrier Sentence

The carrier sentence needs to contain metal as subject of react. There are 34 such
sentences in UKWaC. The next task is to choose the most suitable for a language-teaching, cloze exercise context.

Many sentences are unsuitable, for a range of reasons. For example:

\[ 2 \text{H}_2 \text{O}_2(\text{aq}) \rightarrow 2 \text{H}_2 \text{O}(\text{l}) + \text{O}_2(\text{g}) \]

or a metal reacting with acids, and you can study the effects of a catalyst e.g. adding \( \text{Cu}^{2+}(\text{aq}) \) ions to a zinc-acid mixture, though I’m not sure easy it is to get good quantitative results for advanced level coursework?

Firstly, the sentence is too long, giving the learner work to do which is not directly related to the task that the exercise assesses. Secondly, it contains formulae which will be incomprehensible to non-chemists. Another example is:

It uses these reactions to explore the trend in reactivity in Group 1. The Facts

Here, the problem is that we have not one sentence but two, and a heading and sub-heading in between. The corpus processing module responsible for identifying sentences has been led astray by the period following the 1, interpreting it as part of the token “1.” rather than as an end-of-sentence marker. It has also failed to mark off the heading (“The facts”) and sub-heading (“General”) as not being part of the following sentence.

Atkins and Rundell (2008) discuss the criteria for good examples in dictionary definitions, concluding that such examples must be intelligible to learners, avoiding difficult lexis and structures, puzzling or distracting names, and anaphoric references which cannot be understood without access to the wider context. These lexicographical desiderata are equally applicable to cloze exercises. The SkE concordancing software is equipped with a feature called GDEX (Good Dictionary Example Extraction: Kilgarriff et al, 2008), which ranks sentences extracted from corpora according to the following criteria:

- **Sentence length**: a sentence between 10 and 25 words long is preferred, with longer and shorter ones penalized. (Very short sentences may not provide enough context to show the user the intended meaning of constituent words.)
- **Word frequencies**: a sentence is penalized for each word that is not amongst the commonest 17,000 words in the language, with a further penalty applied for rare words.
- **Sentences containing pronouns and anaphors like *this, that, it* or *one* often fail to present a self-contained piece of language which makes sense without further context, so sentences containing these words are penalized.
- **Sentences where the target collocation is in the main clause are preferred** (using heuristics to guess where the main clause begins and ends, as we do not yet use a parser).
- **Whole sentences, identified as beginning with a capital letter and ending with a full stop, exclamation mark, or question mark, are preferred.**
- **Sentences with ‘third collocates’, that is, words that occurred with high salience in sentences containing the key and KOC, are preferred.** This will increase the chances that the context in which the collocation occurs are typical for the collocation.
- **Sentences with more than two or three capital letters, and more than two or three punctuation marks and other non-alphanumeric characters, are penalized.** This turns out to be a simple way of setting aside most aberrant and junk-filled ‘sentences’.

GDEX sorts the concordance lines for any SkE search so that the ‘best’ sentences are presented first, so the sentences which are most likely to be selected for dictionary examples or cloze exercises appear conveniently at the beginning of the concordance display.
Unwanted sentences, including web noise, are relegated to the end of the concordance so a human user need not waste time looking at them.

TEDDCLOG uses GDEX to find the ‘best’ sentence containing the key+KOC collocation, here metal as subject of react.

**Implementation**

The prototype implementation is in Java. All interactions with the Sketch Engine are over the web, via the web API, as described at [http://trac.sketchengine.co.uk/wiki/SkE/Methods/index](http://trac.sketchengine.co.uk/wiki/SkE/Methods/index). The web API is a recent development and TEDDCLOG is one of the most extensive uses of it to date.

**System Output and Evaluation**

To test and make a preliminary evaluation of the system, a random sample of 80 words from the CEEC list\(^1\) were entered into the system as the cloze item key. The corpus used was the BNC. One carrier sentence and three distractors were output for each key. The first author of the paper, a native speaker of English and experienced language teacher, then assessed whether each item was acceptable or not, and classified the cases where it was not.

Of the 80 items generated, seven were acceptable:

1. Top prize is a day at a recording ____.
   - (a) theatre
   - (b) flat
   - (c) workshop
   - (d) studio

2. The comprehensive on-screen manual will teach novice users how to use the more commonly ____ features.
   - (a) used
   - (b) sophisticated
   - (c) standard
   - (d) accustomed

3. They walk down to the river, as a motley crowd ____ about the jetties where the ferry boats left for the West Bank.
   - (a) mope
   - (b) potter
   - (c) prowl
   - (d) mill

4. Several metals ____ violently with cold water.
   - (a) behave
   - (b) react
   - (c) proceed
   - (d) respond

5. Ten years ago, they didn't take seriously the risk of being sued for ____ smoking.
   - (a) passive
   - (b) arbitrary
   - (c) abstract
   - (d) static

\(^1\) A glossary of 6480 words used to help people studying for university entrance exams (see College Entrance Examination Center 2002).
6. By the road from the ferry to Raasay House, small neat houses appear, bright ___ gardens, attentively tended, gleaming in the sunshine.
   (a) colourful
   (b) lively
   (c) fascinating
   (d) striking

7. The hundred shares ___ closed down three points at twenty-five, sixty-two.
   (a) volume
   (b) score
   (c) index
   (d) list

The above seven items could all be correctly answered from general knowledge of the gapped word. 29 generated items were rejected because the knowledge required to answer them was felt to be too specialist or obscure. The ability to answer (8), (9), (10) and (11), for example, rests on knowledge of the idiomatic *lame duck, optical illusion, plum pudding* and *mushy peas*.

8. It may be desirable to spend what could otherwise be dole money on temporarily subsidizing ___ ducks to ease the transition.
   (a) patronizing
   (b) pathetic
   (c) feeble
   (d) lame

9. The grey ash cone on which we crouched pared away beneath us to blend with the leaden sea like an optical ___.
   (a) dream
   (b) myth
   (c) impression
   (d) illusion

10. Two hours later we clatter down the stairs of a West End restaurant feeling like two ___ puddings on legs.
    (a) pear
    (b) plum
    (c) peach
    (d) strawberry

11. They emerge from the tunnel like an ad for mushy ___.
    (a) bean
    (b) cabbage
    (c) pea
    (d) tomato

18 items were rejected because at least one of the distractors was acceptable in the sentence. Of these 18 items, six offered one infelicitously correct distractor, three offered two such distractors, and in nine cases all the distractors proposed were potentially correct, for example:

12. Office worker Bryan Johnson managed to knock Mr Chittenden to the floor, tearing off his own shirt to ___ the flames.
    (a) extinguish
    (b) stifle
    (c) smother
    (d) douse
It seems that all the distractors are just as likely to occur as the key smother. This is the kind of case which will be resolved when we move to UKWaC, since then the chances of all of extinguish/stifle/douse flames having frequency zero in 1.5 billion words is low. Indeed a check on UKWaC showed that extinguish flame had 148 hits there.

If a distractor begins with a capital letter (that is, the corpus lemma is capitalized), or if it not found in a standard dictionary, it should not be used. See, for example, (13), where the distractors Koi and orfe should have been ruled out.

13. Within broad uncertainties, the explosions of the Kincardine and College objects are consistent with the fates of long-period ___ with kinetic energies of tens of kilotons.
   (a) Koi
   (b) goldfish
   (c) comet
   (d) orfe

Finally, some generated items were ruled out simply because they were too easy:

14. To flap their ___ they will need more than 1,000 square centimetres.
   (a) wing
   (b) arm
   (c) tail
   (d) leg

**Future Work and Summary**

In the immediate future, we shall address the issue with inflection noted above. Rather than presenting lemma forms of the key and distractors to the test-taker, we shall generate and present the correct inflected forms.

Future experiments will use UKWaC not BNC.

The system will be amended to exclude KOCs and distractors which are capitalised or are not common English words.

There are various improvements planned for the GDEX module. We aim to include a language model, so that sentences that are not plausible sentences of standard English are excluded. This should improve the quality of carrier sentences.

We plan to integrate the system into the core Sketch Engine, to improve speed and maintainability.

Evaluation of the prototype system, as noted above, was carried out by one individual. We will shortly assemble a team of evaluators, including ELT professionals and proficiency testing experts, as well as language learners. We also aim to acquire a set of cloze items prepared in the standard way by professional item-writers. We will then mix them with automatically-generated items and ask evaluators to judge which were generated by human, which by machine. We hope to undertake evaluations of this sort working with a language-testing organisation, with whom we can collaborate on turning TEDDCLOG from a research system into a production system.

**Summary**

We have described a program for generates cloze exercises from a corpus. The input is the key (the correct answer) and the system selects distractors and a carrier sentence, such that the distractors are, in some sense, plausible alternatives to the key, but do not work in the carrier sentence. The system uses the Sketch Engine corpus query system: it takes distractors from the distributional thesaurus compiled in the Sketch Engine, and uses the Sketch Engine “Good Dictionary Example” module for selecting carrier sentences. Thus it
generates cloze items which are good examples of natural, authentic English, as well as distractors which will appear, to many students, to be plausible correct answers, but which are in fact incorrect in the context given. Using a very large corpus and methods from computational linguistics, TEDDLOG offers the prospect of making the preparation of cloze items - currently labour-intensive and expensive - much faster and cheaper.

References


