<http://www.diva-portal.org>

Preprint

This is the submitted version of a paper presented at *CLEF 2017 Conference and Labs of the Evaluation Forum*.

Citation for the original published paper:

Parks, M., Karlgren, J., Stymne, S. (2017)

Plausibility Testing for Lexical Resources.

In: *Proceedings of CLEF 2017: Information Access Evaluation meets Multilinguality, Multimodality, and Visualization*

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-212183>

Plausibility Testing for Lexical Resources

Magdalena Parks¹ and Jussi Karlgren² and Sara Stymne¹

¹ Uppsala University

² Gavagai

Abstract. This paper describes principles for evaluation metrics for lexical components and an implementation of them based on requirements from practical information systems.

1 Evaluating information system components

The performance of a component in a complex processing pipeline can influence the function of downstream components, meaning that end-to-end testing also must be performed on entire systems, using approaches based on use cases with target notions that validate the function of the system for the purpose it is built, such as many of the evaluation measures formulated in workshops at CLEF. But a task-based evaluation does not reveal the performance of individual components. Evaluation of knowledge-based components in an information system should be done systematically, ideally in ways which are similar to unit tests done for other technical components, motivated by the need for a development and maintenance team to:

1. know that the component does what is expected of it;
2. support modular design with clean interfaces between modules and simplify refactoring and other development tasks;
3. support cross-project portability of modules;
4. enable bug tracking, optimization, and error analyses in case of system failure or performance dips.

Evaluating learning lexical resources is challenging for several reasons [1].

Firstly, operationalizable intrinsic measures for knowledge based models risk being irrelevant for system performance or measure outcome, rather than learning process. Secondly, hands-off evaluation which takes a general view of component performance and disregards the body of knowledge which it has been designed to address risks missing important aspects of what a component is intended to do. Thirdly, a purely declarative test may be applied in several different ways, some of which may not provide guidance for system improvement.

Typical evaluation resources are based on a list of selected probe terms and target terms which are in some identified relation to those target terms. Examples of synonym tests are the TOEFL test [2], WordSimilarity-353 [3], SIMLEX 999 [4], ConceptSim [5] and so forth, with a more fine-grained approach for probing several semantic relations in BLESS [6]. These resources are reasonable for testing a static resource and are designed to be a sample of general language usage.

2 Requirements for a habitable evaluation scheme

The reasons an information system needs a learning lexical resource (rather than a stable dictionary) can be varied. One central motivation is to help an information system select the most reasonable term from a set of candidates. This is useful in many tasks: in generation, speech recognition, and various variant suggestion situations, both to rank alternatives in the face of choice and to provide defaults in the absence of information. To do this well a lexical resource must have some sense of context. A tentatively useful approach is to use the Cloze procedure, and our suggested test shares many of the starting points with it [7]. In this paper we suggest an approach for testing a lexical resource which is based on *semantic coherence* and allows for topic or domain tailoring.

An evaluation method must not be biased towards any particular kinds of representation since the ultimate goal of an evaluation would be to fairly compare the different knowledge representations. However, the form of the semantic representation can vary greatly from system to system, making a more direct evaluation very difficult. A more universal method of evaluating semantic representations is to design a task which can only be accurately performed given an accurate knowledge of word semantics. Such a task must meet certain criteria.

First, it is essential that the task should be able to be reliably performed using semantic knowledge. Next, it is important that the task should not be able to be reliably performed without that semantic knowledge. More specifically, syntax should not play a key role in a system's ability to perform the task. While it may be impossible to fully eliminate the role of all non-semantic information on the solvability of the task, it is necessary to minimize the impact as much as possible.

Last, it is necessary for the task to have a solution that can be confirmed without the use of semantic knowledge. Using existing semantic knowledge to confirm the solution would bias the test towards systems similar to the semantic knowledge system used. While using a human-generated answer key as a gold standard would be an acceptable solution in terms of correctness, it is highly inefficient if we ever hope to expand the test to various domains or to include different vocabulary, as we would need to have humans carefully go through and create a new gold standard for any new test cases. Ideally, we would like to use a task which can be programmatically generated, including an answer key, while still requiring semantic knowledge to solve.

3 Plausible Utterances

The proposed task is a *plausible utterances* task. To generate this task, we create a set of sentences containing a number of sentences which occurred naturally and have remained unaltered, as well as a number of sentences which occurred naturally but have had one or more words within them swapped for some other word. We call this set a *coconut*, as per examples given by Karlgren et al [1]. The task is then to sort these sentences in order of likelihood of having occurred

naturally. We can then evaluate the ordering of these results to see whether the system reliably ranks the real sentences higher than the fake sentences.

The plausible utterances test can meet the criteria stated above. By replacing words in the sentence, we are creating a sentence which contains words that are, in all likelihood, in discord with the rest of the sentence. As long as semantics plays some role in this, semantic knowledge can be used to perform the task, satisfying our first criteria. Additionally, we can satisfy the second criteria as long as we carefully select word replacements in such a way as to minimize syntactic incongruity. This test approaches language in a more natural setting than e.g. the *word intrusion* task suggested by Chang et al [8], where human assessors were asked to pick out a randomly introduced topical term in a set of topically consistent terms, and is related to the notion of *perplexity*, used to evaluate consistency and coverage language models for e.g. speech recognition [9]; in contrast with perplexity measures which are intrinsic to a data set, our suggested coconut items are transparent and comprehensible to any human language user for e.g. error analysis purposes.

Finally, since we have programmatically generated the coconut, we know which sentences have been manipulated, and which ones have not, so we naturally have an answer key. While it is possible that some manipulated sentences may, by chance, be reasonably plausible, generating and testing with multiple different coconuts should reduce the impact of these false positives.

4 Method of fake sentence generation

The key method involved in generating this plausible utterances task is the generation of the fake sentences. We do this by selecting words from naturally occurring sentences and replacing them with other, syntactically similar words.

It is necessary that our swapping methods minimize the syntactic incongruity of the sentence. To this end, we make use of POS tags as part of our swapping methodology. Specifically, we elect to only replace words which have the same POS. This easily results in sentences which have minimal syntactic incongruity, while adding only the additional requirement of using tagged data.

We use the Brown corpus [10] as our source of naturally occurring sentences, and limit our experiments to swapping only singular nouns, with the tag NN.

While the vast majority of words are simply swapped with any word of the same POS, we find that some specific instances are not covered by the POS tagset. For instance, if we take the natural noun phrase **a dog**, we might end up with **a apple**, which is syntactically incorrect. We correct for this by altering the determiner based on the first letter of the replacement word, a simple correction that works in most cases.

5 Swapping Strategies

Choices concerning the selection of words to be replaced, as well as their replacements, can have a profound impact on the difficulty level of the coconut and can

present different challenges to the semantic knowledge systems being tested. Two variants of the coconut are proposed here.

5.1 Word coconut

The first variant involves the selection of a probe word. A number of sentences are selected which naturally contain the probe word, and a number of sentences are altered to contain the word. To generate these fake sentences, we randomly select real sentences from the Brown corpus [10] which contain some word that has the same part of speech as our probe word. We then swap that word with our probe word. Probe words are randomly selected out of all words that we see occur in more than one sentence, to avoid unique words from showing up in our tests. In this variation, the sentences each contain the same word in different contexts, and the task is to determine which contexts are the most likely to have occurred naturally.

Examples (1) show an example of a word coconut that could be generated using the probe word **dog** using the POS tag **NN**, such that one real sentence is selected and one fake sentence is generated. Note that while both sentences syntactically work, the first sentence would make sense semantically while the second sentence is nonsensical.

- (1) a. The woman walked the dog to the park.
- b. He sipped his dog as he read the newspaper.

Examples (2), on the other hand, show a coconut which may be generated using the same parameters, but is semantically ambiguous. While our methodology may generate such coconuts, we can expect that in most cases there will be relatively few ambiguous sentence pairs within a coconut. For this reason, we should not expect even the best system to be able to solve this task perfectly in all cases, but we can expect it to perform comparatively better given the same coconuts.

- (2) a. The dog sat on the bed.
- b. The old woman saw a dog.

5.2 Sentence coconut

The second variant involves the selection of a sentence and a target word in that sentence. That target word is then replaced with various words, and each variant is used as part of the test set. We randomly select a sentence, similar to our process for the word coconut. In this variation, the sentences are all identical except for the target word. The task is to determine which target word is most likely to have occurred naturally. Examples (3) show a sentence coconut.

- (3) a. The woman walked the dog to the park.
- b. The woman walked the idea to the park.

Ideally, we wish to be able to compare a system’s performance on the different coconut types relative to each other. For this reason, we choose to generate coconuts of both types containing an equal number of sentences. In our experiments, we use eight sentences in each coconut.

Next, we observe that the sentence coconut generation method results in only a single real sentence, while word coconuts may have more. Again, in the interest of comparability, we choose to make word coconuts conform to the sentence coconut, only retrieving a single real usage of the given word for use in the coconut. In the end, we have both word coconuts and sentence coconuts each with a total of eight sentences, one of which is a natural utterance.

6 Evaluating the evaluation

To test the evaluation we created two naïve n-gram language models from the Reuters corpus [11] using the SRILM toolkit [12], using an additive smoothing of 0.1. First, we use the n-gram data to measure the likelihood of an entire probe sentence. Secondly, more specifically, we take a window around the swap word of the probe sentence, based on the range on the n-gram model in question. To test a 3-gram model, we use a 5 word window, centered on the swapped word. The idea here is not to test the sequences in the probe sentence we know are parts of real utterances: by using this context window, we are isolating the n-grams involved in the part of the sentence that we are unsure is a real utterance. We do not expect either of these implementations to perform our task particularly well, but intend to use them to demonstrate certain characteristics of our coconut test.

We run our experiment using, as described above, coconuts of size 8, each with only a single unmodified sentence. We use 2-gram, 3-gram, and 4-gram models for each test. We generate 46 coconuts of each type, and for each type run our experiment using the full context and the context window. For each coconut, we find the ranking of the only unmodified sentence, with the best being 1 and the worst being 8. We calculate the average rank over all coconuts for our final results, shown in table 1.

	Context Window Word Coconut	Full Context Word Coconut	Context Window Sentence Coconut	Full Context Sentence Coconut
2-gram	3.93	3.43	4.85	5.15
3-gram	3.85	3.37	5.24	4.83
4-gram	3.83	3.37	4.91	4.72

Table 1. Results of Coconut size 8, using ngram models, with different implementations

7 Conclusion

We see that sentence coconuts are more resilient to changes in the methodology used to perform the plausible utterances task when compared to word coconuts.

While the results for the word coconuts quickly settle on results which favor the full context implementation, we see that the sentence coconuts do not reliably demonstrate such a pattern. This is an indication that the result is less dependent on the task implementation than the language model itself. Contrary to expectation, we see that the implementation trained on a narrow and thus more specific context window is outperformed on the word coconuts by the implementation trained on entire sentences as context. This is most likely due to data scarcity, and illustrates well that word coconuts, being a less exact model of performance, are more sensitive to implementation details than sentence coconuts. We find that sentence coconuts are recommended as an evaluation metric, in that they provide a better resolution between test models. The coconut tests are currently being introduced in an industrial setting as part of a test suite for testing a learning lexical resource deployed in numerous languages.

References

1. Karlgren, J., Callin, J., Collins-Thompson, K., Gyllensten, A.C., Ekgren, A., Jurgens, D., Korhonen, A., Olsson, F., Sahlgren, M., Schütze, H.: Evaluating learning language representations. In: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer (2015) 254–260
2. Landauer, T., Dumais, S.: A solution to Plato’s problem: The Latent Semantic Analysis theory of acquisition, induction and representation of knowledge. *Psychological Review* **104**(2) (1997) 211–240
3. Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., Ruppín, E.: Placing search in context: The concept revisited. In: Proceedings of the 10th international conference on World Wide Web, ACM (2001)
4. Hill, F., Reichart, R., Korhonen, A.: Simlex-999: Evaluating semantic models with (genuine) similarity estimation. arXiv preprint arXiv:1408.3456 (2014)
5. Schwartz, H.A., Gomez, F.: Evaluating semantic metrics on tasks of concept similarity. In: Proceedings of FLAIRS. (2011)
6. Baroni, M., Lenci, A.: How we BLESSed distributional semantic evaluation. In: Proceedings of the 2011 Workshop on GEometrical Models of Natural Language Semantics, ACL (2011) 1–10
7. Taylor, W.L.: “Cloze procedure”: a new tool for measuring readability. *Journalism Quarterly* **30**(4) (1953) 415–433
8. Chang, J., Boyd-Graber, J.L., Gerrish, S., Wang, C., Blei, D.M.: Reading tea leaves: How humans interpret topic models. In: Proceedings of NIPS. (2009)
9. Katz, S.: Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE transactions on acoustics, speech, and signal processing* **35**(3) (1987) 400–401
10. Francis, W.N., Kucera, H.: Brown corpus manual. Brown University **15** (1979)
11. Lewis, D.D.: Reuters-21578, distribution 1.0 (1997)
12. Stolcke, A., et al.: SRILM – an extensible language modeling toolkit. In: Inter-speech. Volume 2002. (2002) 2002