Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 171 (2020) 2310–2315

Procedia Computer Science

www.elsevier.com/locate/procedia

Third International Conference on Computing and Network Communications (CoCoNet'19)

Towards a Semantic Approach for Candidate Answer Generation in Solving Crossword Puzzles

Anu Thomas[a], Sangeetha S.[b]

[a]*Text Analytics & Natural Language Processing Lab, Department of Computer Applications, National Institute of Technology, Tiruchirappalli,Tamil Nadu, India*

[b]*Text Analytics & Natural Language Processing Lab, Department of Computer Applications, National Institute of Technology, Tiruchirappalli, Tamil Nadu, India*

Abstract

Language games represent one of the most fascinating challenges of research in Artificial Intelligence. One of the problems that lies in this set is the development of an automated solution generator for Crossword puzzles. Automatic resolution of crossword puzzles is an open natural language challenge that demands the filling of puzzle grid with candidate answers, while satisfying the grid constraints. Moreover, providing an accurate list of answer candidates has a major impact on the performance of the automatic crossword resolution task. This paper proposes a semantic-based search approach for generating candidate answer lists for definition type clues by taking advantage of the lexical relations encoded in WordNet, the lexical database for English.

Keywords: Crossword Puzzles; Candidate Answer Generation; Semantic-based Approach; WordNet; Semantic Similarity; Natural Language Processing Application

1. Introduction

Crossword Puzzles (CP) are the most popular word games played across the globe. Resolving CPs is a natural language task as many clues include puns, or nuances. Additionally, CPs demand the filling of puzzle grid with candidate answers, along with satisfying the grid constraints. Hence, completing a crossword puzzle is a formidable natural language task for both humans and computers. However, developing an automatic CP resolver facilitates quick

* Anu Thomas. Tel.: +91-944-772-7937.
E-mail address: anugraha707@gmail.com

10.1016/j.procs.2020.04.250

and easy verification of the accuracy of crossword puzzles when publishing CPs on a large scale, in the form of puzzle book, magazine, or newspaper.

In theory, while solving a Crossword Puzzle, the reader will be supplied with three basic clues namely, partial definition or hint for a word, the starting alphabet for the word and the length of the word. Automatic CP resolution is a two-step process, where the first step is to generate the list of candidate answers and second step is to fill the puzzle grid with suitable candidate answers, satisfying the grid constraints. Moreover, the performance of the automatic crossword resolver depends on the quality of the candidate answer lists provided. Existing systems for automatic CP resolution depends on several different resources for generating candidate answer lists namely, clue databases, dictionaries, Wikipedia titles, online thesaurus and most widely, the database of previously solved CPs [7, 5, 1, 4, 3].This paper introduces a semantic-based approach to generate a list of candidate answers for crossword puzzle grids using a publicly available lexical resource, called WordNet[6]. The proposed architecture measures the semantic similarity between each of the definitions in WordNet and the input clue definitions of the crossword puzzle. Based on this similarity and further using length constraints, candidate answers are generated.

The main contributions of the paper are as follows:

- The proposed methodology makes use of WordNet to generate candidate answers.To the best of our knowledge, this is the first work which makes use of WordNet and semantic similarity measures to generate candidate answer list in solving crossword puzzle grids.
- A similarity metric is used for computing the similarity between a WordNet definition on one side, and the crossword puzzle clue definition on the other.
- The approach also aims to improve the overall execution time by adopting part of speech (POS) tags.

The structure of the paper is as follows: Section 2 describes the related works in this area. Section 3 introduces the proposed methodology followed by results and discussions.

2. Related Work

Automatic CP solution has attained considerable interest in recent times. The main approaches used for this are described as follows:

2.1. Proverb

The early system designed to solve CPs is Proverb [5]. The system depends on knowledge-specific expert modules and crossword database of great dimensions for generating candidate answers along with their confidence scores. Later, for each clue, the weighted candidate lists from various expert modules are merged into a single probabilistic list in order to place them in to the puzzle grid.

2.2. WebCrow

WebCrow [3] creates candidate answer lists from two major sources: Web documents and previously solved crosswords. Using Web Search Module (WSM), the system extracts and filters potential terms from web documents into candidate answer lists. In addition to this, the system creates answer lists by retrieving clues from the databases (DB) of previously solved CPs.WebCrow uses standard database techniques i.e., SQL Full-Text query for retrieving clues similar to the input clue. Later, the system builds a single merged list of all answers along with their associated probabilities. Followed by, filling of puzzle grid with the best of candidate answers using a probabilistic CSP approach similar to Proverb system.

2.3. Dr. FILL

Dr.Fill system [4] solves CP as a Weighted Constraint Satisfaction Problem (WCSP) after extracting candidate answers from various resources namely, collection of previously published puzzles, clue databases, dictionaries, Wikipedia titles, and online thesaurus. Grammatical information (namely, part of speech and root forms) of around 154,000 words is collected from the data provided by the WordNet project.

2.4. SACRY

Unlike WebCrow, SACRY [1] employs state-of-the-art Information Retrieval (IR) techniques to retrieve the clues similar to the input clue from the DB of previously solved CPs. Followed by the re-ranking of clues based on its syntactic structure. Finally, an aggregation algorithm for generating the list of unique candidate answers. Later, these candidate answers are fed into an automatic CP solver, specifically WebCrow.

2.5. CRUCIFORM

CRUCIFORM [7] designed different components (namely, Lucene and Specialized Natural Language Processing(NLP) components) to generate candidate answers for each clue. The system compiles a list of candidate answers along with confidence scores, after merging the answers from different components. Lucene components handle clues in a generic way. Specialized NLP components are designed to handle clues of certain forms like missing last names, missing first names, acronyms, synonyms, hypernyms etc. The task of each component is to recognize a specific clue type, understand the clue and provide the ranked list of candidate answers.

The input to the Lucene component is the list of all clues in the puzzle along with the clue ID and length of the correct answer. For candidate answer generation, the clues are converted into queries to search through the datasets of known clues as well as Wikipedia titles. Finally the output includes the list of candidate answers for each clue, along with the clue ID and the score indicating the quality of the answer. These answers are then used to formulate a CSP solved by the algorithm similar to Dr. Fill.

Previous work by [3] clearly suggests that providing the crossword solver with an accurate list of answer candidates is a critical step in the CP resolution task. For generating candidate answer list, existing systems mainly rely on data available from the Web, Wikipedia, dictionaries, apart from DBs of previously solved CPs. Existing systems such as Cruciform has designed different NLP components for generating candidate answer lists based on clue types. Similar to Cruciform system, we propose an NLP component that performs semantic-based search in generating candidate answer lists for definition type clues by relying upon WordNet lexical relations as well as semantic similarity measures.

3. Proposed Methodology

In the proposed architecture (refer Fig.1), we follow a semantic based searching approach to find out matching definitions from WordNet[6].The whole procedure is explained in algorithm 1,where the SIM function computes the similarity between each clue definitions and WordNet definitions(Def.) by calculating the Resnik similarity score [8] between the component word senses based on the Information Content of the Least Common Subsumer (LCS) (Refer equation1).

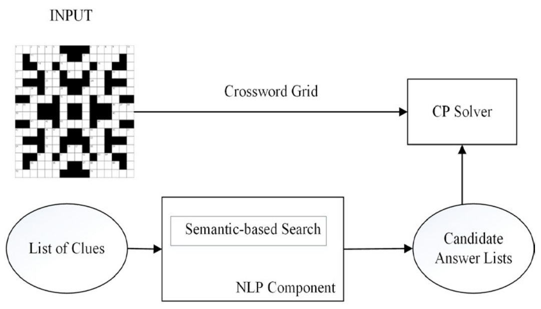

Fig. 1. Proposed Architecture

.

Algorithm 1 Semantic-based Search (SS)

Input:

```
Clue_List  :List({Clue, Length}), list of clues and length of each clue
θ          :Similarity Threshold
WN_Def.    :Definitions from WordNet
SIM        :Function computing semantic similarity between texts
```

Output:

```
Candidate_List  :List of answers corresponding to each clue
```

```
1:  for each multi-word Clue ∈ Clue_List do
2:    Candidate_List ← ∅
3:    L ← EmptyList()
4:    $\beta$ ← SIM(clue ∈ Clue_List, Def. ∈ WN_Def.)
5:    if $\beta \geq \theta$ then
6:      W ← Word corresponding to Def.
7:      S ← Synonyms for each word ∈ W
8:      L ← (W,S)
9:    end-if
10:   if item ∈ L satisfies Length then
11:     Candidate_List ← item
12:   end-if
13: end for
14: return Candidate_List
```

$$SIM(Clue, Def.) = \sum_{i=1}^{|Clue|} \sum_{j=1}^{|Def.|} -logP(LCS(w_i, w_j)) \tag{1}$$

In equation 1 the POS tags of (w_i) and (w_j) should be the same and LCS(w_i,w_j) means Least Common Subsumer of two senses, w_i an dw_j.

To calculate the semantic similarity, the SIM function applies POS tagging (Using NLTK POS tagger[2]) on each input clue definitions and WordNet definitions followed by stopwords removal followed by finding out the similarity between each tokens, one from the input clue and the other from the wordnet definition. Similarity is calculated by measuring the semantic closeness between the synsets associated with each token in the pair. By adopting POS tagging, the synsets corresponding to the particular part of speech are only loaded for each token. This in turn reduces the number of token-wise comparisons and finally improves the computational time.

The WordNet definitions $WN_{Def.}$ whose SIM value is greater than or equal to the given threshold θ (here, we set θ as 6) are selected as candidate definitions. The words that are mapped to each candidate definition are selected as candidate answers for each clue, provided they satisfy the length constraint. Algorithm 1 finally outputs a list of candidate answers corresponding to each clue.

As an example, for the multi-word clue **"a person who takes photographs, especially as a job (7)"**, the proposed algorithm carries out definition searches as opposed to word searches on WordNet. For a given threshold θ, the similarity function SIM selects **"someone who takes photographs professionally"** as a candidate definition, followed by collecting the word corresponding to the candidate definition i.e., **"Photographer"** and its synonym **"Lensman"** into a list. From the list, the algorithm, selects **"Lensman"** as the candidate answer satisfying the length constraint.

4. Results and Discussions

The proposed system that facilitates semantic-based candidate answer generation for crossword puzzles is developed in Python language. The experiments were conducted on an Intel Core(TM) i7-7700 CPU @ 3.60GHz processor based system that uses Ubuntu 16.04 LTS operating system, and has 16GB of main memory.

The proposed methodology was tested on a sample of 100 clues collected from the clue database[1].Table 1 depicts the overall results obtained on testing our approach(includes both with and without POS tagging).

Table 1. Results of the Proposed Method

#Clues	**Average Number of tokens in Clue Definition**	**#Clues Correctly Answered**
100	4	80

Table 2 represents the results obtained for a sample of clue definitions where the text in bold face in candidate answer list represents the clue answer.

Table 2. Results on Sample data

Clue Answer	**Clue Definition**	**Candidate Answer List**
Biology	the science that studies living organisms	{ habitat, ecology, **biology** }
Church	place of worship for Christians	{temple, novena, **church**, mosque, rosary, shrine, masjid, chapel}
Entrepreneur	someone who organizes a business venture	{ big_business, banking_game, organization, **entrepreneur**}
Leisure	time available for ease and relaxation	{ relaxer, **leisure**, gym rat, respite}
Key	Lock opener	{ **key**}
Thief	one who steals	{rifle, purse,**thief**}
Mirror	it reflects images	{ **mirror**}
bug	Computer glitch	{**bug**}
Alcohol	Sour substance that is sharp in taste	{vinegar, Marsala, **alcohol**, liqueur, whiskey, Pilsner, glycine, tequila}

Based on the results, the following are our main findings:

- Proposed methodology used Resnik similarity measure to find out the similarity between two word senses. Resnik measure uses both corpus-based and taxonomy-based information content and so it led to better results for most of the input queries.
- Finding out the POS tag of tokens both in the input clue as well as in the candidate definitions reduced the number of comparisons in computing the semantic similarity between tokens. If POS tag of a token is specified, synsets corresponding to a particular part of speech only will be loaded and compared and this in turn improved the execution time of the algorithm(Refer figure 2).

[1] https://www.crosswordgiant.com

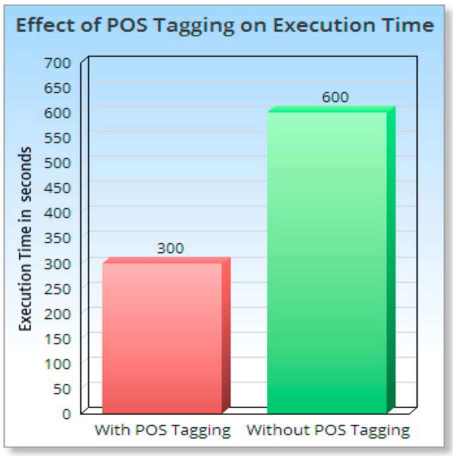

Fig. 2. Influence of POS Tagging on Execution Time

- It is observed that the length of the clue definition has an impact on the execution time since each token in the clue has to be compared with the tokens in each WordNet definitions. Here, in the input sample set, the number of tokens in the clue definitions lies in the range of 2-6 (average is 4) resulting in computational time of ≈300 and ≈600 seconds with and without POS tagging respectively(base don experimental results). Execution time of the algorithm increases with the number of tokens in the clue definitions.
- The results in table 1, the proposed methodology fails to answer 20 out of 100 input clue definitions. This is mainly because of the fact that clue definitions contain only few tokens that are able to convey proper meaning of the whole sentence even after removing stop words. This happens in clue definitions of the form **’cause to stop’**, **’inhabit or live in’**.

5. Conclusion

In this paper, we proposed a semantic-based search approach that automatically generates candidate answers for crossword puzzle grids. Semantic-based search extends the usability of WordNet and semantic similarity measures in generating candidate answers and thus it overcomes the drawbacks of traditional lexicon-based search in existing systems. Based on the preliminary results, we conclude that incorporating WordNet could be a viable solution to the problem, but we need to experiment more on this to improve the results. Our next step will be on incorporating word embedding techniques as well as word sense disambiguation techniques to handle different types of clues.

References

[1] Barlacchi, G., Nicosia, M., Moschitti, A., 2015. Sacry: Syntax-based automatic crossword puzzle resolution system. ACL-IJCNLP 2015 , 79.
[2] Bird, S., Loper, E., 2004. Nltk: The natural language toolkit, in: Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions, Association for Computational Linguistics, Stroudsburg, PA, USA. URL: http://dx.doi.org/10.3115/1219044.1219075, doi:10.3115/1219044.1219075.
[3] Ernandes, M., Angelini, G., Gori, M., 2005. Webcrow: A web-based system for crossword solving, in: Proceedings of the 20th National Conference on Artificial Intelligence - Volume 3, pp. 1412–1417.
[4] Ginsberg, M.L., 2011. Dr. fill: Crosswords and an implemented solver for singly weighted csps. Journal of Artificial Intelligence Research 42, 851–886.
[5] Littman, M.L., Keim, G.A., Shazeer, N., 2002. A probabilistic approach to solving crossword puzzles. Artificial Intelligence 134, 23–55.
[6] Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J., 1990. Introduction to wordnet: An on-line lexical database. International journal of lexicography 3, 235–244.
[7] Radev, D., Zhang, R., Wilson, S., Van Assche, D., Gubert, H.S., Krivokapic, A., Dong, M., Wu, C., Bondera, S., Brandl, L., et al., 2016. Cruciform: Solving crosswords with natural language processing. arXiv preprint arXiv:1611.02360 .
[8] Resnik, P., 1995. Using information content to evaluate semantic similarity, in: Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp. 448–453.