AI-KU: Paper Title

First Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Second Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

ID	Token
R2	how
R2	blog
R2	popular

Table 2: Input example for AI-KU$_1$.

Abstract

1 Introduction

2 Algorithm

In this section, we will explain preprocessing steps of the data and the details of our two participated systems[1] for this task. Both systems rely on the co-occurrence statistics. The slight differences between the first and the second system is that the first one uses the words occur in the text (e.g., paragraph, sentence etc.), whereas the second system employs co-occurrence statistics on 100 most likely substitutes of each occurred word in text.

2.1 Data Preprocessing

Two AI-KU systems distinguishes between each other on input of co-occurrence modeling part.

AI-KU$_1$: This system uses the words that observed in the text. All words are translated into lower-case equivalents. Lemmatization[2], stop-word removing[3] has been done and finally non-alphanumeric characters are excluded. Table 2 shows the input of the AI-KU$_1$ for the following sentence which is an instance from paragraph2sentence test set:

How can my blog be more popular?

[1]The code to replicate our work can be found *input github repo link here*.

[2]Stanford CoreNLP is used for lemmatization.

[3]Python NLTK library is used.

	System	Pearson	Spearman
Paragraph-2-Sentence	Best	0.837	0.821
	2^{nd} Best	0.834	0.820
	3^{rd} Best	0.826	0.817
	AI-KU$_1$	0.732	0.727
	AI-KU$_2$	0.698	0.700
	LCS	0.527	0.613
	lch	0.125	0.114
	jcn	0.107	0.106
	JI	0.640	0.687

Table 3: Paragraph-2-Sentence subtask scores for the test set. *Best* indicates the best score for the subtasks. LCS stands for Normalized Longest Common Substring. Subscripts in AI-KU systems indicate the run number.

AI-KU$_2$: Unlike the first system, this system represents each context of a word by finding the most likely substitutes suggested by the 4-gram language model we built from ukWaC[4] (Ferraresi et al., 2008), a 2-billion word web-gathered corpus. This high probability substitutes work successfully on Word Sense Induction (WSI) *cite semeval 2013* and Part-of-Speech Induction *Mali & Enis work* problems. We used the 100 most likely substitute for each context. No lemmatization, stop word removing and lower-case transformation has been done. Table 1 illustrates the context and substitutes of each context using a bigram language model.

2.2 Co-Occurrence Modeling

This subsection will explain the unsupervised methods we employed to model co-occurrence statistics: the Co-occurrence data Embedding (CODE) method (Globerson et al., 2007) and its

[4]Available here: http://wacky.sslmit.unibo.it

Word	Context	Substitutes
the	<s> ___ dog	w_1 (0.01), w_2 (0.53), w_3 (0.13), ..., w_n (0.02)
dog	the ___ bites	w_1 (0.01), w_2 (0.1), w_3 (0.05), ..., w_n (0.01)
bites	dog ___ </s>	w_1 (0.03), w_2 (0.23), w_3 (0.04), ..., w_n (0.01)

Table 1: Contexts when using a bigram language model

	System	Pearson	Spearman
Sentence-2-Phrase	Best	0.777	0.642
	2^{nd} Best	0.771	0.760
	3^{rd} Best	0.760	0.757
	AI-KU$_1$	0.680	0.646
	AI-KU$_2$	0.617	0.612
	LCS	0.562	0.626
	lch	0.420	0.431
	jcn	0.469	0.535
	JI	0.540	0.555

Table 4: Sentence2phrase subtask scores for the test set. *Best* indicates the best score for the subtasks. LCS stands for Normalized Longest Common Substring. Subscripts in AI-KU systems indicate the run number.

spherical extension (S-CODE) proposed by Maron et al. (2010). Unlike our previous WSI work (Baskaya et al., 2013) where we ended up with an embedding for each word in the co-occurrence modeling step, in this task however, we try to model each text unit such as a paragraph, a sentence or a phrase, thus, after this step, we obtain embeddings for each paragraph, sentence and phrase instance.

Input data for S-CODE algorithm were instance-id of the text unit and the each word in the text unit for the first system, the 100 most likely substitutes of each word in text for the second system. In the initial step, S-CODE puts all instance-ids and words/substitutes randomly on n-dimensional sphere. If two different instances have the same word/substitute, then these two instances attract each other, otherwise they will repel. When S-CODE converges, instances have similar words/substitutes will be closely located, instances have no similar words/substitutes will be far from each other.

2.3 Similarity Calculation

When the S-CODE converges, we have an n-dimensional embedding for each textual level (e.g., paragraph, sentence, phrase) instance. We can use a similarity metric to calculate the similarity between embeddings. For this task, systems should report only the similarity between two specific cross level instances. Note that we used cosine similarity in order to calculate similarity between two textual unit. *How about the other metrics? Make a test and input the reason why we use cosine similarity*. This similarity is the eventual similarity for two instances; no further processing (e.g., scaling) has been done.

3 Evaluation Results

In this task, systems were evaluated with two correlation metrics; Pearson correlation and Spearman's rank correlation. Pearson correlation tests the degree of similarity between the system's similarity ratings and the gold standard ratings. Spearman's rho tests the degree of similarity between the rankings of the items according to similarity.

Table 3 and 4 show the scores for Paragraph-2-Sentence and Sentence-2-Phrase subtasks, respectively. These tables contain the best individual scores for the performance metrics, Normalized Longest Common Substring (LCS) baseline which is given by task organizers and three additional baselines: lch (Leacock and Chodorow, 1998), jcn (Jiang and Conrath, 1997), and Jaccard Index (JI) baseline. jcn uses information content (Resnik, 1995) of the least common subsumer (LCS) of concepts A and B. Information content indicates the specificity of a concept and LCS of a concept A and B is the most specific concept from which A and B inherited. *Paraphrase: jcn measure augments the information content of the LCS with the sum of the information content of concepts A and B themselves and takes the difference of this sum and the information content of the LCS.* On the other hand, *Paraphrase: lch is based*

on path lengths between a pair of concepts. It finds the shortest path between two concepts, and scales that value by the maximum path length found in the isa hierarchy in which they occur (please see Pedersen et al. (2004) for further details of these measures). These two baselines were calculated as follows. First, using the Stanford Part-of-Speech Tagger (Toutanova and Manning, 2000) we tagged words in all textual levels. After tagging, we found the synsets of each word matched with its part-of-speech using WordNet 3.0 (Fellbaum, 1998). For each synset of a word in the shorter textual unit (e.g., sentence is shorter than paragraph), we calculated the jcn/lch measure of each synset of all words in the longer textual unit and picked the highest score. When we found the scores for all words, we sum them up and normalized it with the length of the shorter textual unit. This gave us the similarity between one pair in the test set. Finally, Jaccard Index baseline simply calculates the number of words in common (intersection) with two cross textual levels, normalized by the total number of words (union).

Both AI-KU systems outperformed the Normalized Longest Common Substring (LCS) baseline.

4 Conclusion

Acknowledgements

References

Osman Baskaya, Enis Sert, Volkan Cirik, and Deniz Yuret. 2013. Ai-ku: Using substitute vectors and co-occurrence modeling for word sense induction and disambiguation. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 300–306.

Christine Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukwac, a very large web-derived corpus of english. In *In Proceedings of the 4th Web as Corpus Workshop (WAC-4*.

Amir Globerson, Gal Chechik, Fernando Pereira, and Naftali Tishby. 2007. Euclidean embedding of co-occurrence data. *Journal of Machine Learning Research*, 8(10).

Jay J Jiang and David W Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. *arXiv preprint cmp-lg/9709008*.

C. Leacock and M. Chodorow. 1998. Combining local context and wordnet similarity for word sense identification. In Christiane Fellfaum, editor, *MIT Press*, pages 265–283, Cambridge, Massachusetts.

Yariv Maron, Michael Lamar, and Elie Bienenstock. 2010. Sphere Embedding: An Application to Part-of-Speech Induction. In J Lafferty, C K I Williams, J Shawe-Taylor, R S Zemel, and A Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1567–1575.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet:: Similarity: measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, pages 38–41. Association for Computational Linguistics.

Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*.

Kristina Toutanova and Christopher D Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 13*, pages 63–70. Association for Computational Linguistics.