MathLingBudapest: Concept Networks for Semantic Similarity

Gábor Recski
Research Institute for Linguistics
Hungarian Academy of Sciences
Benczúr u. 33
1068 Budapest, Hungary
recski@mokk.bme.hu

Judit Ács
HAS Research Institute for Linguistics and
Dept. of Automation and Applied Informatics
Budapest U of Technology and Economics
Magyar tudósok krt. 2 (Bldg. Q.)
1117 Budapest, Hungary
judit@mokk.bme.hu

Abstract

We present our approach to measuring semantic similarity of sentence pairs used in Semeval 2015 tasks 1 and 2. We adopt the sentence alignment framework of (Han et al., 2013) and experiment with several measures of word similarity. We hybridize the common vector-based models with definition graphs from the 4lang concept dictionary and devise a measure of graph similarity that yields good results on training data. We did not address the specific challenges posed by Twitter data, and this is reflected in placing 11th from 30 in Task 1, but our systems perform fairly well on the generic datasets of Task 2, with the hybrid approach placing 11th among 78 runs.

1 Introduction

This paper describes the systems participating in Semeval-2015 Task 1 (Xu et al., 2015) and Task 2 (Agirre et al., 2015). To compute the semantic similarity of two sentences we use the architecture presented in (Han et al., 2013) to find, for each word, its counterpart in the other sentence that is semantically most similar to it. We implemented several methods for measuring word similarity, among them (i) a word embedding created by the method presented in (Mikolov et al., 2013) and (ii) a metric based on networks of concepts derived from the 4lang concept lexicon (Kornai and Makrai, 2013; Kornai et al., 2015) and definitions from the Longman Dictionary of Contemporary English (Bullon, 2003). A hybrid system exploiting both of these metrics yields the best results and placed 11th among 73 systems on Semeval Task 2a (Semantic Textual Similarity for English). All components of our system are available for download under an MIT license from GitHub[1][2]. Section 2 describes the system architecture and points out the main differences between our system and that in (Han et al., 2013), section 3 outlines our word similarity metric derived from the 4lang concept lexicon. We present the evaluation of our systems on both tasks in section 4, and section 5 provides a brief conclusion.

2 Architecture

Our framework for determining the semantic similarity of two sentences is based on the system presented in (Han et al., 2013). Their architecture, *Align and Penalize*, involves computing an alignment score between two sentences based on some measure of word similarity. We've chosen to reimplement this system so that we can experiment with various notions of word similarity, including the one based on 4lang and presented in section 3. Although we reimplemented virtually all rules and components described by (Han et al., 2013) for experimentation, we shall only describe those that ended up in at least one of the five configurations submitted to Semeval.

The core idea behind the *Align and Penalize* architecture is, given two senteces S_1 and S_2 and some measure of word similarity, to align each word of one sentence with some word of the other sentence so that the similarity of word pairs is maximized.

[1] http://github.com/juditacs/semeval
[2] http://github.com/kornai/pymachine

Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), pages 138–142,
Denver, Colorado, June 4-5, 2015. ©2015 Association for Computational Linguistics

The mapping need not be one-to-one and is calculated independently for words of S_1 (aligning them with words from S_2) and words of S_2 (aligning them with words from S_1). The score of an alignment is the sum of the similarities of each word pair in the alignment, normalized by sentence length. The final score assigned to a pair of sentences is the average of the alignment scores for each sentence. For out-of-vocabulary (OOV) words, i.e. those that are not covered by the component used for measuring word similarity, we use the Dice-similarity over the sets of character 4-grams in each word. Additionally, we use simple rules to detect acronyms and compounds: if a word of one sentence that is a sequence of 2-5 characters (e.g. *ABC*) has a matching sequence of words in the other sentence (e.g. *American Broadcasting Company*), all words of the phrase are aligned with this word and recieve an alignment score of 1. If a sentence contains a sequence of two words (e.g. *long term* or *can not*) that appear in the other sentence without a space and with or without a hyphen (e.g. *long-term* or *cannot*), these are also aligned with a score of 1. The score returned by the word similarity component can be boosted based on WordNet (Miller, 1995), e.g. if one is a hypernym of the other, if one appears frequently in glosses of the other, or if they are derivationally related. For the exact cases covered and a description of how the boost is calculated, the reader is referred to (Han et al., 2013). In our submissions we only used this boost on word similarity scores obtained from word embeddings.

The similarity score may be reduced by a variety of penalties, which we only enabled for Task 1 runs – they haven't brought any improvement on Task 2 datasets in any of our early experiments. Of the penalties described in (Han et al., 2013) we only used the one which decreases alignment scores if the word similarity score for some word pair is very small (< 0.05). We also introduced two new types of penalties based on our observations of error types in Twitter data: if one sentence starts with a question word and the other one does not or if one sentence contains a past-tense verb and the other does not, we reduce the overall score by $1/(L(S_1) + L(S_2))$, where $L(S_1)$ and $L(S_2)$ are the numbers of words in each sentence.

3 Similarity from Concept Networks

This section will present the word similarity measure based on principles of lexical semantics presented in (Kornai, 2010). The `4lang` concept dictionary (Kornai and Makrai, 2013) contains 3500 definitions created manually. Because the Longman Defining Vocabulary (LDV) (Boguraev and Briscoe, 1989) is a subset of `4lang`, we could automatically extend this manually created seed to every headword of the Longman Dictionary of Contemporary English (LDOCE) by processing their definitions with the Stanford Dependency Parser (Klein and Manning, 2003), and mapping dependency relations to sets of edges in the `4lang`-style concept graph. Details of the mapping will be described elsewhere (Recski, 2015).

Since these definitions are essentially graphs of concepts, we have experimented with similarity functions over pairs of such graphs that capture semantic similarity of the concepts defined by each of them. There are two fundamentally different configurations present in `4lang` graphs:

1. two nodes may be connected via a 0-edge, which is a generalization over unary predication ($\texttt{dog} \xrightarrow{0} \texttt{bark}$), attribution ($\texttt{dog} \xrightarrow{0} \texttt{faithful}$), and hypernymy, or the `IS_A` relation ($\texttt{dog} \xrightarrow{0} \texttt{mammal}$).

2. two nodes can be connected, via a 1-edge and a 2-edge respectively, to a third one representing a binary relation. Binaries include all transitive verbs (e.g. $\texttt{cat} \xleftarrow{1} \texttt{CATCH} \xrightarrow{2} \texttt{branch}$). and a handful of binary primitives (e.g. $\texttt{tree} \xleftarrow{1} \texttt{HAS} \xrightarrow{2} \texttt{branch}$).

We start by the intuition that similar concepts will overlap in the elementary configurations they take part in: they might share a 0-neighbor, e.g. $\texttt{train} \xrightarrow{0} \texttt{vehicle} \xleftarrow{0} \texttt{car}$, or they might be on the same path of 1- and 2-edges, e.g. $\texttt{park} \xleftarrow{1} \texttt{IN} \xrightarrow{2} \texttt{town}$ and $\texttt{street} \xleftarrow{1} \texttt{IN} \xrightarrow{2} \texttt{town}$.

We'll define the *predicates* of a node as the set of such configurations it takes part in. For example, based on the definition graph in Figure 1, the predicates of the concept

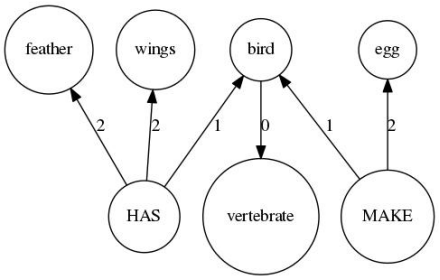

Figure 1: 4lang definition of `bird`.

`bird` are {`vertebrate; (HAS, feather); (HAS, wing); (MAKE, egg)`}.

Our initial version of graph similarity is the Jaccard similarity of the sets of predicates of each concept, i.e.

$$S(w_1, w_2) = \frac{|P(w_1) \cap P(w_2)|}{|P(w_1) \cup P(w_2)|}$$

For all words that are not among the 3500 defined in `4lang` we obtain definition graphs by automated parsing of Longman definitions and the application of a simple mapping from dependency relations to graph edges (Recski, 2015). By far the largest source of noise in these graphs is that currently there is no postprocessor component that recognizes common structures of dictionary definitions like appositive relative clauses. For example the word `casualty` is defined by LDOCE as *someone who is hurt or killed in an accident or war* and we currently build the graph in Figure 2 instead of that in Figure 3. To mitigate the effects of these anomalities, we updated our definition of predicates: we let them be "inherited" via paths of 0-edges encoding the `IS_A`-relationship.

We've also experimented with similarity measures that take into account the sets of all nodes accessible from each concept in their respective definition graphs. This proved useful in establishing that two concepts which would otherwise be treated as entirely dissimilar are in fact somewhat related. For example, given the definitions of the concepts `casualty` and `army` in Figures 2 and 4, the node `war` will allow us to assign nonzero similarity to the pair `(army, casualty)`. We found it most effective to use the maximum of these two types of similarity.

Testing several versions of graph similarity on

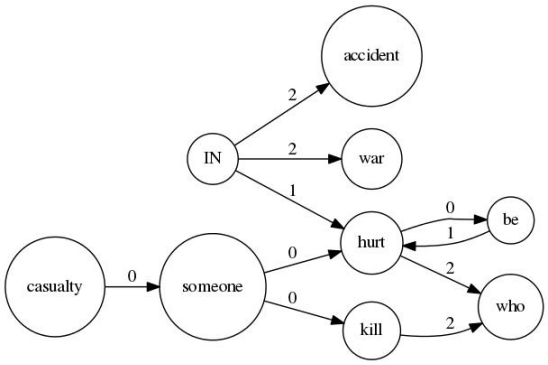

Figure 2: Definition of `casualty` built from LDOCE.

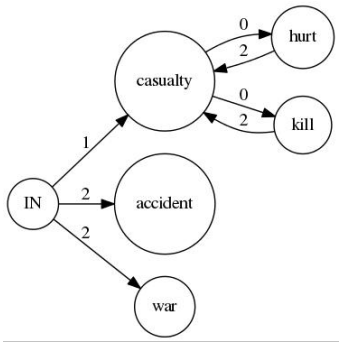

Figure 3: Expected definition of `casualty`.

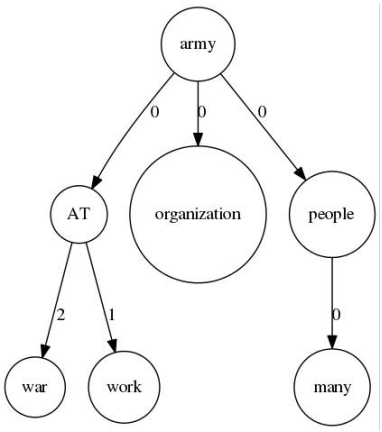

Figure 4: Definition of `army` in `4lang`.

past years' STS data, we found that if two words w_1 and w_2 are connected by a path of 0-edges, it is best to treat them as synonymous, i.e. assign to them a similarity of 1. This proved very efficient for determining semantic similarity of the most common types of sentence pairs in the Semeval datasets. Two descriptions of the same event (common in the *headlines* dataset) or the same picture (in *images*) will often only differ in their choice of words or choice of concreteness. In a dataset from 2014, for example, two descriptions, likely of the same picture, are *A bird holding on to a metal gate* and *A multi-colored bird clings to a wire fence*. Similarly, a pair of news headlines are *Piers Morgan questioned by police* and *Piers Morgan Interviewed by Police*. Although *wire* is by no means a synonym for *metal*, nor does being *questioned* mean exactly the same as being *interviewed*, treating them as perfect synonyms proved to be an efficient strategy when trying to assign sentence similarity scores that correlate highly with human annotators' judgements.

4 Submissions

We shall now describe the particular configurations used for each submission in Semeval. For Task 1 (Paraphrase and Semantic Similarity in Twitter) we ran two systems: `twitter-embed` uses a single source of word similarity, a word embedding built from a corpus of word 6-grams from the Rovereto Twitter N-Gram Corpus[3] using the `gensim`[4] package's implementation of the method presented in (Mikolov et al., 2013). Our second submission, `twitter-mash` combines several sources of word similarity by averaging the output of various systems using weights that have been learned using plain least squares regression on the training data available. The systems participating in the vote differ in the word similarity measure they use: one subset uses the character ngram baseline described in section 2 with various parameters ($n = 2, 3, 4$, each with Jaccard- and Dice-similarity), two systems use word embeddings (built from 5-grams and 6-grams of the Rovereto corpus, respectively) and one uses the `4lang`-based word similarity described in section 3.

	`embedding`	`hybrid`
Task 1a: *Paraphrase Identification*		
Precision	0.454	0.364
Recall	0.594	0.880
F-score	**0.515**	**0.515**
Task 1b: *Semantic Similarity*		
Pearson	0.229	**0.511**

Table 1: Performance of submitted systems on Task 1.

	`embedding`	`machine`	`hybrid`
Task 2a: *Semantic Similarity*			
answers-forums	0.704	0.698	0.723
answers-students	0.700	0.746	0.751
belief	0.733	0.736	0.747
headlines	0.769	0.805	0.804
images	0.804	0.841	0.844
mean Pearson	0.748	0.777	**0.784**

Table 2: Performance of submitted systems on Task 2.

For Task 2 (Semantic Textual Similarity) we were allowed three submissions. The `embedding` system uses a word embedding built from the first 1 billion words of the English Wikipedia using the `word2vec`[5] tool (Mikolov et al., 2013). The `machine` system uses the word similarity measure described in section 3 (both systems use the character ngram baseline as a fallback for OOVs). Finally, for the `hybrid` submission we used a weighted sum of these two systems and the character ngram baseline (weights were once again obtained using simple least square regression on the available training data). In both hybrid submissions we trained on a single dataset consisting of all training data available, we haven't experimented with genre-specific models.

Our results on each task are presented in Tables 1 and 2. In case of Task 1a (Paraphrase Identification) our two systems performed equally in terms of F-score and ranked 30th among 38 systems. On Task 1b the hybrid system performed considerably better than the purely vector-based run, placing 11th out of 28 runs. On Task 2 our hybrid system ranked 11th among 78 systems, the systems using the word embedding and the `4lang`-based similarity alone (with string similarity as a fallback for OOVs in each case) ranked 22nd and 15th, respectively.

[3] `http://clic.cimec.unitn.it/amac/twitter_ngram/`

[4] `http://radimrehurek.com/gensim`

[5] `https://code.google.com/p/word2vec/`

5 Conclusion

In a framework like (Han et al., 2013) which approximates sentence similarity by word similarity, the first order of business is to get the word similarity right. Character ngrams are quite useful for this, and remain an invaluable fallback even when more complex measures of word similarity, such as embeddings, are used. Dictionary-based methods, such as the `4lang`-based system presented here, are slightly better, and require only a one-time investment of manual labor to generate the seed. Critically, the error characteristics of the context-based (embedding) and the dictionary-based systems are quite different, so hybridizing the two provides a real boost over both.

Acknowledgments

Recski designed and implemented the machine-based similarity, Ács reimplemented the align-and-penalize architecture and created the word embedding. We thank András Kornai (HAS Institute for Computer Science) and Márton Sipos (Budapest U of Technology) for useful comments and discussions.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 Task 2: Semantic Textual Similarity, English, Spanish and Pilot on Interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, Denver, CO, U.S.A.

Branimir K. Boguraev and Edward J. Briscoe. 1989. *Computational Lexicography for Natural Language Processing*. Longman.

Stephen Bullon. 2003. *Longman Dictionary of Contemporary English 4*. Longman.

Lushan Han, Abhay Kashyap, Tim Finin, James Mayfield, and Jonathan Weese. 2013. UMBC_EBIQUITY-CORE: Semantic textual similarity systems. In *Proceedings of the 2nd Joint Conf. on Lexical and Computational Semantics*, pages 44–52.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430.

András Kornai and Márton Makrai. 2013. A 4lang fogalmi szótár. In Attila Tanács and Veronika Vincze, editors, *IX. Magyar Számitógépes Nyelvészeti Konferencia*, pages 62–70.

András Kornai, Judit Ács, Márton Makrai, Dávid Nemeskey, Katalin Pajkossy, and Gábor Recski. 2015. Competence in lexical semantics. To appear in Proc. *SEM-2015.

András Kornai. 2010. The algebra of lexical semantics. In Christian Ebert, Gerhard Jäger, and Jens Michaelis, editors, *Proceedings of the 11th Mathematics of Language Workshop*, LNAI 6149, pages 174–199. Springer.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In Y. Bengio and Y. LeCun, editors, *Proc. ICLR 2013*.

George A. Miller. 1995. Wordnet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Gábor Recski. 2015. Building concept graphs from monolingual dictionary entries. Unpublished manuscript.

Wei Xu, Chris Callison-Burch, and William B. Dolan. 2015. SemEval-2015 Task 1: Paraphrase and semantic similarity in Twitter (PIT). In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval)*.