CLaC-CORE: Exhaustive Feature Combination for Measuring Textual Similarity

Ehsan Shareghi
CLaC Laboratory
Concordia University
Montreal, QC H3G 1M8, CANADA
eh_share@cse.concordia.ca

Sabine Bergler
CLaC Laboratory
Concordia University
Montreal, QC H3G 1M8, CANADA
bergler@cse.concordia.ca

Abstract

CLaC-CORE, an exhaustive feature combination system ranked 4th among 34 teams in the Semantic Textual Similarity shared task STS 2013. Using a core set of 11 lexical features of the most basic kind, it uses a support vector regressor which uses a combination of these lexical features to train a model for predicting similarity between sentences in a two phase method, which in turn uses all combinations of the features in the feature space and trains separate models based on each combination. Then it creates a meta-feature space and trains a final model based on that. This two step process improves the results achieved by single-layer standard learning methodology over the same simple features. We analyze the correlation of feature combinations with the data sets over which they are effective.

1 Introduction

The Semantic Textual Similarity (STS) shared task aims to find a unified way of measuring similarity between sentences. In fact, sentence similarity is a core element of tasks trying to establish how two pieces of text are related, such as Textual Entailment (RTE) (Dagan et al., 2006), and Paraphrase Recognition (Dolan et al., 2004). The STS shared task was introduced for SemEval-2012 and was selected as its first shared task. Similar in spirit, STS differs from the well-known RTE shared tasks in two important points: it defines a graded similarity scale to measure similarity of two texts, instead of RTE's binary yes/no decision and the similarity relation is considered to be symmetrical, whereas the entailment relation of RTE is inherently unidirectional.

The leading systems in the 2012 competition used a variety of very simple lexical features. Each system combines a different set of related features. CLaC Labs investigated the different combination possibilities of these simple lexical features and measured their performance on the different data sets. Originally conceived to explore the space of all possible feature combinations for 'feature combination selection', a two-step method emerged that deliberately compiles and trains all feature combinations exhaustively and then trains an SVM regressor using all combination models as its input features. It turns out that this technique is not nearly as prohibitive as imagined and achieves statistically significant improvements over the alternative of feature selection or of using any one single combination individually.

We propose the method as a viable approach when the characteristics of the data are not well understood and no satisfactory training set is available.

2 Related Work

Recently, systems started to approach measuring similarity by combining different resources and methods. For example, the STS-2012 shared task's leading UKP (Bär et al., 2012) system uses n-grams, string similarity, WordNet, and ESA, and a regressor. In addition, they use MOSES, a statistical machine translation system (Koehn et al., 2007), to translate each English sentence into Dutch, German, and Spanish and back into English in an effort to increase their training set of similar text pairs.

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task*, pages 202–206, Atlanta, Georgia, June 13-14, 2013.

TakeLab (Šaric et al., 2012), in place two of the 2012 STS shared task, uses n-gram models, two WordNet-based measures, LSA, and dependencies to align subject-verb-object predicate structures. Including named-entities and number matching in the feature space improved performance of their support vector regressor.

(Shareghi and Bergler, 2013) illustrates two experiments with STS-2012 training and test sets using the basic core features of these systems, outperforming the STS-2012 task's highest ranking systems. The STS-2013 submission CLaC-CORE uses the same two-step approach.

3 CLaC Methodology

Preprocessing consists of tokenizing, lemmatizing, sentence splitting, and part of speech (POS) tagging. We extract two main categories of lexical features: explicit and implicit.

3.1 Explicit Lexical Features

Sentence similarity at the explicit level is based solely on the input text and measures the similarity between two sentences either by using an n-gram model (*ROUGE-1, ROUGE-2, ROUGE-SU4*) or by reverting to string similarity (longest common subsequence, *jaro, ROUGE-W*):

Longest Common Subsequence (Allison and Trevor, 1986) compare the length of the longest sequence of characters, not necessarily consecutive ones, in order to detect similarities

Jaro (Jaro, 1989) identifies spelling variation between two inputs based on the occurrence of common characters between two text segments at a certain distance

ROUGE-W (Lin et al., 2004a), a weighted version of *longest common subsequence*, takes into account the number of the consecutive characters in each match, giving higher score for those matches that have larger number of consecutive characters in common. This metric was developed to measure the similarity between machine generated text summaries and a manually generated gold standard

ROUGE-1 unigrams (Lin et al., 2004a)

ROUGE-2 bigrams (Lin et al., 2004a)

ROUGE-SU4 4-Skip bigrams (including Unigrams) (Lin et al., 2004a)

3.2 Implicit Lexical Features

Sentence similarity at the implicit level uses external resources to make up for the lexical gaps that go otherwise undetected at the explicit level. The synonymy of *bag* and *suitcase* is an example of an implicit similarity. This type of implicit similarity can be detected using knowledge sources such as WordNet or Roget's Thesaurus based on the WordNet::Similarity package (Pedersen et al., 2004) and combination techniques (Mihalcea et al., 2006). For the more semantically challenging non-ontologigal relations, for example *sanction* and *Iran*, which lexica do not provide, co-occurrence-based measures like ESA are more robust. We use:

Lin (Lin, 1998) uses the Brown Corpus of American English to calculate information content of two concepts' least common subsumer. Then he scales it using the sum of the information content of the compared concepts

Jiang-Conrath (Jiang and Conrath, 1997) uses the conditional probability of encountering a concept given an instance of its parent to calculate the information content. Then they define the distance between two concepts to be the sum of the difference between the information content of each of the two given concepts and their least common subsumer

Roget's Thesaurus is another lexical resource and is based on well-crafted concept classification and was created by professional lexicographers. It has a nine-level ontology and doesn't have one of the major drawbacks of WordNet, which is lack of links between part of speeches. According to the schema proposed by (Jarmasz and Szpakowicz, 2003) the distance of two terms decreases within the interval of [0,16], as the the common head that subsumes them moves from top to the bottom and becomes more specific. The electronic version of Roget's Thesaurus which was developed by (Jarmasz and Szpakowicz, 2003) was used for extracting this score

Explicit Semantic Analyzer (Gabrilovich and Markovitch, 2007) In order to have broader coverage on word types not represented in lexical resources, specifically for named entities, we add explicit semantic analyzer (ESA) generated features to our feature space

3.3 CLaC-CORE

CLaC-CORE first generates all combinations of the 11 basic features (*jaro*, *Lemma*, *lcsq*, *ROUGE-W*, *ROUGE-1*, *ROUGE-2*, *ROUGE-SU4*, *roget*, *lin*, *jcn*, *esa*), that is $2^{11} - 1 = 2047$ non-empty combinations. The *Two Phase Model Training* step trains a separate Support Vector Regressor (SVR) for each combination creating 2047 *Phase One Models*. These $2^N - 1$ predicted scores per text data item form a new feature vector called *Phase Two Features*, which feed into a SVR to train our *Phase Two Model*.

On a standard 2 core computer with $\leq$100 GB of RAM using multi-threading (thread pool of size 200, a training process per thread) it took roughly 15 hours to train the 2047 Phase One Models on 5342 text pairs and another 17 hours to build the Phase Two Feature Space for the training data. Building the Phase Two Feature Space for the test sets took roughly 7.5 hours for 2250 test pairs.

For the current submissions we combine all training sets into one single training set used in all of our submissions for the STS 2013 task.

4 Analysis of Results

Our three submission for STS-2013 compare a baseline of *Standard Learning* (RUN-1)with two versions of our *Two Phase Learning* (RUN-2, RUN-3). For the *Standard Learning* baseline, one regressor was trained on the training set on all 11 *Basic Features* and tested on the test sets. For the remaining runs the *Two Phase Learning* method was used. All our submissions use the same 11 *Basic Features*. *RUN-2* is our main contribution. *RUN-3* is identical to *RUN-2* except for reducing the number of support vectors and allowing larger training errors in an effort to assess the potential for speedup. This was done by decreasing the value of γ (in the *RBF* kernel) from 0.01 to 0.0001, and decreasing the value of C (error weight) from 1 to 0.01. These parameters resulted in a smoother and simpler decision surface but negatively affected the performance for *RUN-3* as shown in Table 1.

The STS shared task-2013 used the Pearson Correlation Coefficient as the evaluation metric. The results of our experiments are presented in Table 1. The results indicate that the proposed method, *RUN-2*, was successful in improving the results achieved by our baseline *RUN-1* ever so slightly (the confidence invervals at 5% differ to .016 at the upper end) and far exceeds the reduced computation version of *RUN-3*.

	rank	headlines	OnWN	FNWN	SMT
RUN-1	10	0.6774	0.7667	0.3793	0.3068
RUN-2	7	0.6921	0.7367	0.3793	0.3375
RUN-3	46	0.5276	0.6495	0.4158	0.3082
STS-bl	73	0.5399	0.2828	0.2146	0.2861

Table 1: CLaC-CORE runs and STS baseline performance

4.1 Successful Feature Combinations

Having trained separate models based on each subset of features we can use the predicted scores generated by each of these models to calculate their correlations to assess which of the feature combinations were more effective in making predictions and how this most successful combination varies bewteen the different datasets.

	best	**worst**
headlines	[ROUGE-1 ROUGE-SU4 esa lem]	[jcn lem lcsq]
	0.7329	0.3375
OnWN	[ROUGE-1 ROUGE-SU4 esa lin jcn roget lem lcsq ROUGE-W]	[jaro]
	0.7768	0.1425
FNWN	[roget ROUGE-1 ROUGE-SU4]	[ROUGE-2 lem lcsq]
	0.4464	-0.0386
SMT	[lin jcn roget ROUGE-1]	[esa lcsq]
	0.3648	0.2305

Table 2: Best and worst feature combination performance on test set

Table 2 lists the best and worst feature combinations on each test set. *ROUGE-1* (denoted by RO-1), unigram overlap, is part of all four best performing subsets. The features *ROUGE-SU4* and *Roget's*

appear in three of the best four feature combinations, making Roget's the best performing lexicon-based feature outperforming WordNet features on this task. *esa*, *lin*, *jcn* are part of two of the best subsets, where *lin* and *jcn* occur together both times, suggesting synergy. Looking at the worst performing feature combinations is also instructive and suggests that *lcsq* was not an effective feature (despite being at the heart of the more successful *ROUGE-W* measure).

We also analyze performance of individual features over different datasets. Table 3 lists all the features and, instead of looking at only the best combination, takes the top three best combinations for each test and compares how many times each feature has occurred in the resulting 12 combinations (first column). Three clear classes of effectiveness emerge, high (10-7), medium (6-4), and low (3-0). Next, we observe that the test sets differ in the average length of the data: *headlines* and *OnWN* glosses are very short, in contrast to the other two. Table 3 shows in fact contrastive feature behavior for these two categories (denoted by short and long). The last column reports the number of time a feature has occurred in the best combinations (out of 4). Again, *ROUGE-1*, *ROUGE-SU4*, and *roget* prove effective across different test sets. *esa* and *lem* seem most reliable when we deal with short text fragments, while *roget* and *ROUGE-SU4* are most valuable on longer texts. The individual most valuable features overall are *ROUGE-1, ROUGE-SU4*, and *roget*.

Features	total (/12)	short (/6)	long (/6)	best (/4)
esa	6	6	0	2
lin	6	3	3	2
jcn	4	1	3	2
roget	9	3	6	3
lem	6	6	0	2
jaro	0	0	0	0
lcsq	3	3	0	1
ROUGE-W	7	4	3	1
ROUGE-1	10	6	4	4
ROUGE-2	3	1	2	0
ROUGE-SU4	10	5	5	3

Table 3: Feature contribution to the three best results over four datasets

5 Conclusion

CLaC-CORE investigated the performance possibilities of different feature combinations for 11 basic lexical features that are frequently used in semantic distance measures. By exhaustively training all combinations in a two-phase regressor, we were able to establish a few interesting observations.

First, our own baseline of simply training a SVM regressor on all 11 basic features achieves rank 10 and outperforms the baseline used for the shared task. It should probably become the new standard baseline.

Second, our two-phase exhaustive model, while resource intensive, is not at all prohibitive. If the knowledge to pick appropriate features is not available and if not enough training data exists to perform feature selection, the exhaustive method can produce results that outperform our baseline and one that is competitive in the current field (rank 7 of 88 submissions). But more importantly, this method allows us to forensically analyze feature combination behavior contrastively. We were able to establish that unigrams and 4-skip bigrams are most versatile, but surprisingly that Roget's Thesaurus outperforms the two leading WordNet-based distance measures. In addition, *ROUGE-W*, a weighted longest common subsequence algorithm that to our knowledge has not previously been used for similarity measurements shows to be a fairly reliable measure for all data sets, in contrast to longest common subsequence, which is among the lowest performers.

We feel that the insight we gained well justified the expense of our approach.

Acknowledgments

We are grateful to Michelle Khalife and Jona Schuman for their comments and feedback on this work. This work was financially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Adam Kilgarriff, Pavel Rychly, Pavel Smrz, and David Tugwell. 2004. ITRI-04-08 The Sketch Engine. *Information Technology*.

Alex J. Smola and Bernhard Schölkopf. 2004. A Tutorial on Support Vector Regression. *Statistics and Computing*, 14(3).

Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics*, 32(1).

Chin-Yew Lin. 2004a. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*.

Chin-Yew Lin and Franz Josef Och. 2004b. Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics.

Christiane Fellbaum 2010. WordNet. *Theory and Applications of Ontology: Computer Applications*. Springer.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012. UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics*.

Dekang Lin. 1998. An Information-Theoretic Definition of Similarity. In *Proceedings of the 15th International Conference on Machine Learning*, volume 1.

Ehsan Shareghi, Sabine Bergler. 2013. Feature Combination for Sentence Similarity. To appear in *Proceedings of the 26st Conference of the Canadian Society for Computational Studies of Intelligence (Canadian AI'13). Advances in Artificial Intelligence*, Regina, SK, Canada. Springer-Verlag Berlin Heidelberg.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 Task 6: A Pilot on Semantic Textual Similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics*.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*.

Frane Šaric, Goran Glavaš, Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašic. 2012. TakeLab: Systems for Measuring Semantic Text Similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The Pascal Recognising Textual Entailment Challenge. *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*.

Jay J. Jiang and David W. Conrath. 1997. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. *Proceedings of the 10th International Conference on Research on Computational Linguistics*.

Lloyd Allison and Trevor I. Dix. 1986. A Bit-String Longest-Common-Subsequence Algorithm. *Information Processing Letters*, 23(5).

Mario Jarmasz and Stan Szpakowicz. 2003. Rogets Thesaurus and Semantic Similarity. In *Proceedings of the Conference on Recent Advances in Natural Language Processing*.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: an Update. *ACM SIGKDD Explorations Newsletter*, 11(1).

Matthew A. Jaro. 1989. Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida. *Journal of the American Statistical Association*.

Philip Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcelo Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In *Proceedings of the National Conference on Artificial Intelligence*.

Ted Pedersen, Siddharth Patwardhan and Jason Michelizzi. 2004. WordNet:: Similarity: Measuring the Relatedness of Concepts. In *Demonstration Papers at North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

William B. Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources. In *Proceedings of the 20th International Conference on Computational Linguistics*. Association for Computational Linguistics.